The Coolest HyperCard Book Ever!

Cool Mac™ Stacks

David Drucker

Wow!
Disk includes
19 Mac Stacks

"Have fun with this book...I know I did!"
–Robertson Reed Smith
HyperZealot Engineer
Claris Corporation

HAYDEN

Cool Mac Stacks

David Drucker

HAYDEN

Cool Mac Stacks

Copyright © 1992 by Hayden. A Division of Prentice Hall Computer Publishing.

All rights reserved. Printed in the United States of America. No part of this book may be used or reproduced in any form or by any means, or stored in a database or retrieval system, without prior written permission of the publisher except in the case of brief quotations embodied in critical articles and reviews. Making copies of any part of this book for any purpose other than your own personal use is a violation of United States copyright laws. For information, address Hayden, 11711 N. College Ave., Carmel, IN 46032.

Library of Congress Catalog No.: 91-78348

ISBN: 0-672-48522-2

This book and accompanying disk are sold *as is*, without warranty of any kind, either express or implied. While every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information or instructions contained herein. It is further stated that the publisher and author are not responsible for any damage or loss to your data or your equipment that results directly or indirectly from your use of this book or the software accompanying it.

94 93 92 4 3 2

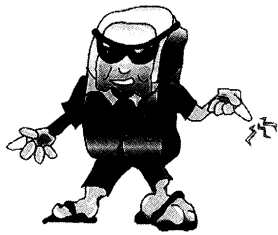
Interpretation of the printing code: the rightmost double-digit number is the year of the book's printing; the rightmost single-digit number the number of the book's printing. For example, a printing code of 92-1 shows that the first printing of the book occurred in 1992.

Cool Mac Stacks is based on HyperCard Version 2.1 and the earlier Version 2.0.

Trademark Acknowledgments: Hayden has made every effort to supply trademark information about company names, products, and services mentioned in this book. Trademarks indicated below were derived from various sources. Hayden cannot attest to the accuracy of this information.

Adobe Illustrator and PhotoShop are trademarks of Adobe Systems Inc. AppleEvents and ResEdit are trademarks, and AppleTalk, HyperCard, HyperTalk, Mac LC, Mac SE/30, Mac IIsi, Mac IIci, Mac, Macintosh, MultiFinder, and the Apple logo are registered trademarks of Apple Computer, Inc. CompuServe is a trademark of H&R Block, Inc. FreeHand is a trademark and PageMaker is a registered trademark of Aldus Corporation. GEnie is a trademark of General Electric Company. MacWrite is a registered trademark of Claris Corporation. Microsoft Word is a registered trademark of Microsoft Corporation. Post-it is a registered trademark of 3M. Quark XPress is a registered trademark of Quark, Inc. Rolodex is a registered trademark of Rolodex Corporation. Super 3D is a trademark of Silicon Beach Software, Inc., a subsidiary of Aldus Corporation. Stacks on the disk were compacted by Compact Pro™ AutoExtractor™© 1991 Bill Goodman.

Dedication



For my parents...

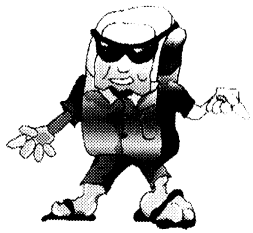
About the Author

David Drucker

David Drucker was originally trained as a composer and conductor, attending the University of Cincinnati's College-Conservatory of Music, Cambridge University, and The Eastman School of Music. During that time, he fiddled around with a Mac and worked as a music critic for the Rochester *Democrat and Chronicle*. After moving to Boston, he started attending The Boston Computer Society Macintosh Group's meetings and soon became Director of Education for BCS•Mac and leader of the HyperCard/MultiMedia Group. Specializing in User Interface Design, Drucker has worked since 1987 as a HyperCard, SuperCard, and Spinnaker PLUS consultant, trainer, and developer. His clients include large and small businesses, as well as schools, hospitals, and the government. He is a frequent speaker at the MacWorld Exposition in the United States and Canada, and has written for *The Active Window* (magazine of BCS•Mac), *MacUser UK*, and *Mac Tech Quarterly*. He lives in Cambridge, Massachusetts with his wife Pam, and two cats, Socrates and Stefi.

Publisher	Mike "Great" Britton
Associate Publisher	Karen "Don't It Make My Brown Eyes" Bluestein
Managing Editor	Neweleen "Cuisine" Trebnik
Production Editor	Tad "Yellow Submarine" Ringo
Copy Editor	Erik "The Viking" Dafforn
Mac Guru	Mel "Tormé" Beckman
Editorial Production Coordinator	Karen "I Want to Live in The" Whitehouse
Cover Designer	Katherine "Pixel Power" Hanley
Designers	Scott "Can't but Wish I Could" Cook Michele "Rodeo Queen" Laseau
Gaffer	Richard "Swadman" Swadley
Key Grip	Matthew "A" Morrill
Best Boy	Jodi "And Her Boy Elroy" Jensen
Costume Designer	Mary-Terese Elizabeth Ann "I Need a Shorter Name" Cozzola Cagnina
Hair & Makeup	Greg "Doberman" Pilcher
Production Stunts	Brad "Not by the Hair of My Chinny-Chin" Chinn, Audra "Milk Chocolate" Hershman, Juli "I Don't Mean Maybe" Pavey, Johnna "On the Loose" VanHoose

Filmed entirely on location in Carmel, Indiana, USA.
Composed in ITC Garamond, Helvetica, and MCPdigital by Prentice Hall Computer Publishing.



Acknowledgments

This book would not exist if it were not for some innovative ideas by Karen Bluestein and Mike Britton of Hayden. Karen and Mike also provided much encouragement and many suggestions, and set me straight on many of the details of the book. Karen Whitehouse added care and consideration. Thanks to all of the team in Carmel, Indiana.

The Boston Computer Society Macintosh Group was an outstanding resource. BCS•Mac remains the user group with not only great quantity, but also quality of volunteers. Thanks go particularly to Mike Feller, who has made the BCS•Mac HyperCard Library the most impressive collection anywhere.

Cool Mac Stacks is the sum of so many people's creativity and effort that it is impossible to list them all. At the very least, here are the authors of the cool stacks included in this collection:

Stack Name	*Author*
Phone->Acronym	Peter Baum
WildAbacus	Mike Atanasio
Sunrise Demo	Leighton Paul
Renaissance Man	A. DeVito
Tangram	Martin Lewis
Bond of Union	Chuck Pratt
HyperConference	Jeff Benjamin
Balloons	HyperActive Software
PICS Player	Kevin Altys
Status Window	Jim Parshall
Colorizing HyperCard	BungDabba Productions
Recipes, About Boxer, Say the Numbers, and The Pizza Navigator	David Drucker
FullHPop, ListSelect, Tabloid, and Textoid	Frédéric Rinaldi

Thanks to all the stackware authors who submitted material, and to those who agreed to let us publish their stacks. For those who did not make it into the final cut, whether for limits of time or disk space, legal issues, or communications, I am sorry. I would have liked to see twice the amount of material included.

Thanks to Robertson Reed Smith for the Foreword. He is a kindred spirit and we are all lucky to have him on the HyperCard team.

Last and most, my wife Pam was a constant help, dealing with contacting the authors, editing some of my manuscript, and keeping me organized, sane, and loved. I could not have done any of this without her.

Conventions Used in This Book

Text that you are instructed to type appears in a `special typeface`.

Text that appears on-screen, such as messages and menu commands, appears in **bold**.

Keystroke combinations are indicated as in the following example: "Press command-P to print." This means you press *and hold* the command key (the key at the bottom of your keyboard with a picture of both an apple and a cloverleaf on it), and then press the key labeled "P". Release both keys simultaneously.

The following icons are used throughout the book:

Icon	*Explanation*
	Cool idea.
	Clue/hint.
	Important note.

Notice

Hayden's licensor(s) makes no warranties, express or implied, including without limitation the implied warranties of merchantability and fitness for a particular purpose, regarding the software. Hayden's licensor(s) does not warrant, guarantee or make any representations regarding the use or the results of the use of the software in terms of its correctness, accuracy, reliability, currentness or otherwise. The entire risk as to the results and performance of the software is assumed by you. The exclusion of implied warranties is not permitted by some states. The above exclusion may not apply to you.

In no event will Hayden's licensor(s), and their directors, officers, employees or agents (collectively Hayden's licensor) be liable to you for any consequential, incidental, or indirect damages (including damages for loss of business profits, business interruption, loss of business information, and the like) arising out of the use or inability to use the software even if Hayden's licensor has been advised of the possibility of such damages. Because some states do not allow the exclusion of or limitation of liability for consequential or incidental damages, the above limitations may not apply to you. Hayden's licensor's liability to you for actual damages from any cause whatsoever, and regardless of the form of the action—whether in contract, tort (including negligence), product liability or otherwise—will be limited to $50.

Foreword

Until the revolution in August of 1987 (otherwise known as the Introduction of HyperCard), the computer "for the rest of us" really belonged to someone else: the members of that sacred elite priesthood—the programmers. With computer science degrees as a rite of passage, programmers dictated what we saw and did with the Macintosh, just as they have with all other computers for decades.

Then came HyperCard, and with it, tens of thousands of scripters looking for information and ways to distribute their stacks. One of the leaders who emerged from this revolution to meet the challenge was David Drucker. Throughout HyperCard's history, David's passion for evangelizing this "software erector set" has been matched only by his commitment to share, instruct, and create with it himself, as I think this book shows.

His dedicated work with the Boston Computer Society brought him to my attention all the way on the West Coast—not only as the source for some of the latest and greatest stacks, but as the focal point for the HyperCard consciousness on the East Coast. Since then, his articles, participation on networks and speaking at computer conventions have shaped the way we think about interactive environments and helped the popularity of HyperCard go far beyond anything anyone on the team ever imagined.

So it comes as no surprise that I find his book is excellent, too. It's well researched and written on a nice, personal level that is easily understood and that won't intimidate anyone. *Cool Mac Stacks* starts out with the basics for the beginner and gradually builds through its examples and stacks until it reaches your skill level. Then it empowers you to go further, because David is one of those rare individuals who believes that every moment you spend interacting with your Mac should stimulate your imagination, excite your senses, reward your efforts, and be uniquely yours. That can only happen in an environment like HyperCard and with tools like the ones offered in this book. So, I would recommend this book to HyperCard users of any skill level and tell them to have fun with it. I know I did.

Robertson Reed Smith
HyperZealot/Engineer
Claris Corporation

Creator of Stack Starter, Power Tools, Train Set, Readymade Buttons, Readymade Fields (coauthor of the Audio Palette, HyperBattleship!), Colorizing HyperCard, and many more stacks.

Disclaimer

Executive summary: We take full credit for anything you like about this product. We designed it just for you. Any aspect of this product that dissatisfies you is purely an accident or simply not our fault.

We at Hayden have tried our best to make this the most fun and useful Macintosh book available to the masses. However, we are (unfortunately) only human, and it is conceivable that you may encounter unusual circumstances that we did not discover in our extensive two-hour testing session. For any problems you may have, we apologize in advance. We'll do our best to make amends wherever possible (and inexpensive).

Be careful. Practice safe computing. Make backups. Try the programs out on a friend's Mac first...

Void where prohibited by law. No stopping, standing, or parking. No shirts, no shoes, no service. Place trays in the upright and locked position. This side up. Do not submerge. Curb your dog. Do not fold, spindle, or mutilate. Tape. Do not staple. Fully microwaveable. Under penalty of law, this tag may not be removed except by the consumer. Do not pass GO, do not collect $200. Have you enclosed your check? No CODs. Keep out of reach of children. Keep off the grass. If product gets in eyes, flush with warm water. No trespassing. This is not a toy. Non-negotiable. Please enclose stub with payment. Open 9 a.m. 'til closing.

We Want to Hear from You

What our readers think of Hayden is crucial to our sense of well-being. If you have any comments, no matter how great or how small, we'd appreciate your taking the time to send us a note, fax us a fax, rhyme us a rhyme, etc.

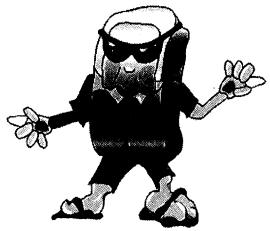
We can be reached at the following address:

Mike Britton/Karen Bluestein
Hayden: The Cool Mac Publisher
11711 N. College Ave. (a boring address, but we assure you we're anything but boring)
Carmel, IN (yes, Indiana—not California...) 46032
(317) 573-2500 voice
(317) 573-2583 fax

If this book has changed your life, please write and describe the euphoria you've experienced. If you have a cool mutant stack of your own creation, please send it (caged, of course) directly to us at Hayden. Who knows? Your stack may end up in *Cool Mac Stacks: The Sequel.*

Do you have a cool book idea? Please contact us at the above address with any proposals (marriage or otherwise).

Table of Contents



Introduction

1 About This Cool Book 1
2 What Is HyperCard? 5
3 HyperCard for the Browser 11
4 Creating HyperCard Stacks 17
5 About External Commands 27

The Stacks

Information Enhancement

6 Recipes: A Tasty Database 31
7 Phone->Acronym: When Numbers Are Not Enough 37
8 WildAbacus 41
9 What Time Is Tomorrow's Sunrise? 45

Fun and Games

10 Renaissance Man: A Word Game with a Twist 49
11 Tangram: An Age-Old Game Goes Digital 53
12 The Pizza Navigator: A Stack to Go 57
13 *Bond of Union:* A Digital Interpretation 61

Utilities

14 Fast Talk, or HyperConference 63
15 Balloons: Inflatable HyperCard Help 69
16 Say the Numbers: The Number You Have Dialed... 75

Building Blocks

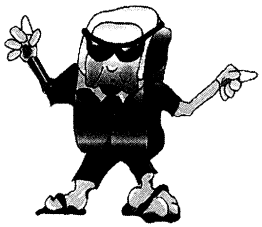
17 PICS Player: A New Way to Get Animation into HyperCard 79
18 A Stack That Answers the Question "How's It Going?" 85
19 About Boxer, for the Egotist in All of Us 89
20 Colorizing HyperCard: A Look at Some Local Color 93
21 FullHPop: Everything's Popping Up Roses 97
22 ListSelect: A Stack for Freedom of Choice 101
23 Tabloid: No, This Is Not a Stack About *The Enquirer* 105
24 Textoid: I Can't Believe It's an XCMD 109

Resources

25 Cool Solutions to Uncool Problems 115
26 On-Line Services and User Groups 125
27 Browsing Quick Reference 127

Index 129

Using the Cool Disk 133
How This Book Was Created 133
What's on the Cool Disk? 134



1

About This Cool Book

Let's face it, you got a Mac because it was cool. Sure, you got it to raise your level of productivity, provide powerful business solutions, calculate, process, file, communicate, and generally make your work and life easier. This is all true, but there is a secret smile that we all get when the Mac does something truly thrilling or even truly goofy. That's what the Cool Mac series is about: the things you can do with your computer to stimulate, amuse, and even enlighten yourself, friends, and coworkers. This series is about all of that. It's about having one of those cool experiences with your Mac, because that's another reason you got a Macintosh.

Computers are unique because they are soft machines. They constantly change shape and function. Simply by running a new application, your Macintosh becomes an entirely new

tool. Your Mac can help you draw diagrams, paint posters, compose music, write novels, design airplanes, print magazines, chart sailing routes, and engage in thousands of other endeavors, depending on which icon you double-click. Much of the world is touched by computers and the Macintosh sitting in front of you is waiting to carry you into it.

The Cool Mac series provides an introduction to many areas of the Mac that were limited to insiders. It invites you to do more with your Mac than just word processing or number crunching. Turn the page, pop the disk in the drive, and dive in.

Why This Book Is for You

If you are completely new to HyperCard, *Cool Mac Stacks* is for you. If you are a beginning or even an intermediate HyperCard user or author, this book is for you, too. Much of what you see in this book and on the accompanying disk demonstrates what HyperCard can do. This book and disk contain HyperCard-based software that you can use right off the disk, and some useful tools and tricks that you can incorporate into your own creations.

The chapters in this book are arranged in a progression from simple, straightforward stacks to involved explorations of the application. Each chapter begins with system and hardware requirements and the level of expertise you should have to best appreciate the chapter's contents. If you follow the stacks in order, you will notice a steady climb from easy-to-use stacks to sophisticated authoring tools.

What's in This Book

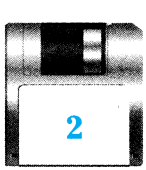
This book is about inspiration. Its purpose is to show inspiring ideas, and hopefully to inspire you, too. *Cool Mac Stacks* is a collection of HyperCard stacks with accompanying descriptions, commentary, and helpful hints. The stacks are examples of what some folks have done with HyperCard, and the examples fall into four categories:

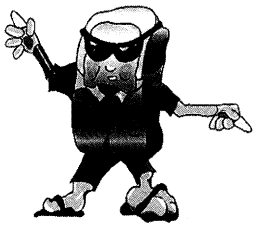
1. Information Enhancement

   HyperCard originally was called a *Personal Information Toolkit*. Stacks can help you organize your life, or calculate and process information in new and interesting ways. These stacks offer unique applications of HyperCard as an informational tool.

2. Fun and Games

   These stacks show that it is possible to create interesting and engaging games and other diversions on the Macintosh, using integrated sound and graphics. You don't have to be a Macintosh programmer to create them.





3. Utilities

   HyperCard stacks can perform some useful functions for your system. The latest version of HyperCard (Version 2.1), in combination with Apple's System 7, can communicate across networks with other copies of HyperCard or other Macintosh applications. HyperCard has always been able to launch other applications and documents, so it also works well as an organizational hub for projects on your computer.

4. Building Blocks

   HyperCard is modular software. You can add new capabilities to stacks with special resources called External Commands (*XCMDs*). Excellent illustrations of this technology are provided to use in your own stacks. In addition, there are some examples of HyperTalk scripting that you can copy and paste into your stacks.

The Cool Stuff on the Disk

HyperCard stacks are made to share. You can share text, graphics, scripts, icons, sounds, and animations between stacks. So, I, along with all the authors of the stacks included here, invite you to dig in and discover how some of these magic tricks are done. Nothing on this disk is password-protected or locked in any way, so feel free to look under the hood of every stack and to copy and try out particularly cool scripts and XCMDs in your own stacks. The best way to learn how to create HyperCard stacks is to see how other people put together their stacks. HyperCard is powerful software, but it is even more powerful when you add what you learn from other stacks. We all stand on the shoulders of others.

What You Need to Run the Stacks on the Disk

To use the example disk, you need a current copy of HyperCard, version 2.0 or later. A few stacks require HyperCard 2.1 and System 7.0, but most require System 6.05 or later. Running HyperCard under System 7.0 requires at least two megabytes of memory, but four is much better. If you bought a System 7.0 Upgrade Kit from Apple, HyperCard 2.1 is included. If you need to get a copy of HyperCard 2.1, you can purchase it from your local dealer or from Claris Corporation. In every case, you need a hard disk. As mentioned before, each stack lists its particular system hardware and software requirements.

Where We Got the Stacks

With the exception of a few exclusives, much of the material here was culled from the collections of User Groups, such as the Boston Computer Society, and from submissions from interested HyperCard developers who use commercial online services, such as America Online, GEnie, and CompuServe, Inc. If you are interested in connecting with any of these sources, Chapter 26, "On-Line Services and User Groups," lists how to get in touch with them.

About Freeware and Shareware

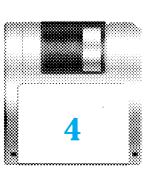
Many of the stacks on this disk are distributed as *freeware*. This means that the author does not expect any payment for the use of the stack or any part of the document. A few stacks are listed as *shareware.* This means the author has requested a nominal payment if you think that their creation is useful to you and that the hard work they put into their stack is worth a contribution (often to finance further updates). Some of the stacks have external commands or scripts that are free if you intend to use them in noncommercial products for yourself, friends, and colleagues. If you intend to create a commercial HyperCard product using resources from one of these stacks, there is information about how to get in touch with the developer to arrange for licensing.

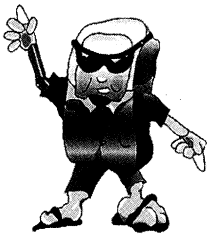
Where You Go from Here

If you are completely new to HyperCard, this book contains an overview of what HyperCard is, information about how to browse through a HyperCard stack, and a short tutorial about how to create your own stacks.

If you have used HyperCard before, you might skip the tutorial and head straight to the stacks. In case you run into a problem, Chapter 25, "Cool Solutions to Uncool Problems," provides troubleshooting advice.

If you received HyperCard 2.1 with a new Mac or with a System 7.0 Upgrade, your version might be set to a lower level of access (called *userLevel*) than HyperCard requires for the stacks in this collection. See the fourth tip in Chapter 25, "Cool Solutions to Uncool Problems," to find out how to reset your userLevel. This is especially important for people who have what is sometimes called HyperCard Lite.





2

What Is HyperCard?

Trying to define HyperCard is a little like the story about the blind men and the elephant. (In case you haven't heard it, several blind men argue about what an elephant actually is. One who touches the animal's tail thinks it is a kind of snake, one who touches the ear thinks it is a bird, and another who touches a foot believes that the elephant is a tree.) Just when you think you have a clear idea of what this software is, someone creates something with HyperCard that doesn't fit your definition. Fortunately, there are three ways to determine what HyperCard is. One is to figure out what it does, another is to see where it comes from, and a third is to see what it consists of.

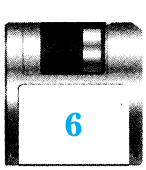
What Does HyperCard Do?

Like a great deal of computer software, HyperCard creates and views documents. These documents, called *stacks*, can contain text, graphics, and sound. Stacks can also include some structure or organization to navigate, retrieve, process, and present this information. What makes HyperCard documents special is the wide variety of media they encompass, as compared to the capabilities of a word processor or database document. For example, one stack can contain both still images and animation; melodies or digitized speech and music; articles; numerical data; and charts. In addition, HyperCard stacks can use the Macintosh computer's calculating capabilities to further enhance this information. If you care to customize the information for yourself, for example, by creating links between material you think is connected or attaching notes to other information, HyperCard provides the tools to do so.

HyperCard blurs the distinction between document and application (or computer program). Sometimes the value of a stack lies in the information it contains, as is clearly the case with a disk-based catalog or on-line help system. Other stacks emphasize the Macintosh's processing power, such as stacks that calculate loans or send network messages. There are even stacks that seem to fall squarely in the middle between the two, containing both data and intelligence. The possibilities of what a HyperCard stack can contain or do are limited only by the imagination and resourcefulness of the stack author.

Where Does HyperCard Come From?

Some of the ideas behind HyperCard were brewed years ago. In 1945, Vannevar Bush, President Roosevelt's Director of the Office of Scientific Research and Development, wrote an article for *Atlantic Monthly* entitled *As We May Think*. In the article, Bush described a hypothetical machine called a *Memex*. The Memex (short for *mem*ory-*ex*tender) was a machine for storing the contents of books, records, and other reference materials on microfilm. Bush envisioned a desk with several translucent screens along with many sets of buttons and levers. The Memex not only let you view all of this information, but also provided lightning-fast leaps from one piece of microfilm to another. You could also save a trail of these links, which Bush felt was an important part of the device's uniqueness and value. To build a Memex, Bush needed a vast amount of storage and a swift retrieval mechanism. As we can see now, although the medium of storage and mechanical details is different, the Macintosh and HyperCard are not far away from Vannevar Bush's original prophecies.



In the mid-1960s a computer science visionary named Theodore Nelson coined the term *hypertext*, which meant a new, revolutionary kind of writing. Hypertext in its most general sense means nonsequential writing. It doesn't need a computer to exist *per se*, but a computer makes hypertext practical in the same way that interstate highways enable us

to better use automobiles. If you don't have to shuffle your papers to go to a linked section of text, hypertext becomes much more useful. The word hypertext did not really come into usage until 1985, nearly 20 years after Nelson had his first ideas about how on-screen writing on computers would change the world. HyperCard lets you link pieces of information together nonsequentially, but because this information can be text, graphics, or sound, the term *hypermedia* would more aptly describe stacks as documents.

HyperCard was first hatched by Bill Atkinson, one of the first employees at Apple Computer. Bill worked on several key pieces of the Apple Lisa and Macintosh operating system software, and wrote one of the first Macintosh applications—MacPaint. MacPaint, along with MacWrite, came with every Macintosh. MacPaint was a black-and-white painting application that was easy enough for computer novices to use, and there had never been anything like it before. In early prototypes, HyperCard was an authoring system with which you could link screens painted with MacPaint-style painting tools. The programming language part—called HyperTalk—came later, with many contributions by Dan Winkler, a brilliant young Apple employee. When HyperCard 1.0 was introduced in 1987, all of these features were integrated into the HyperCard application. The additions of text and sound were developed later as well. As of this publication, revisions and updates to HyperCard have been taken up by other programmers working out of Apple for Claris Corporation, a software subsidiary of Apple.

HyperCard's Dual Personality

HyperCard is software with a split personality. One personality reads or plays documents (again, the idea of the Memex might be helpful here). The other personality builds. HyperCard is best thought of in building or authoring mode as a software erector set for creating stack documents. In authoring mode, you may create graphics, specify links between information, or program HyperTalk scripts. In *browse* mode, you may navigate through documents and perhaps add, change, or delete text information in them. The signal that you are in browse mode is your cursor, which looks like a pointing index finger. The cursor is the only tool for browse mode, but it changes shape if it is located over text that you may edit. In this case, you see the more familiar *I-beam* cursor from Macintosh word processing applications. Figure 2.1 shows the browse tool cursor and I-beam cursor.

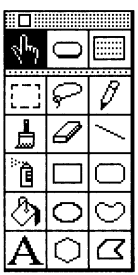
Figure 2.1
The browse tool and I-beam cursors.

To change from browse mode to authoring mode in HyperCard, you must choose a tool from the **Tools** menu. The **Tools** menu tears off to become a tool palette. If you don't see a **Tools** menu when you start up HyperCard, don't panic. Your copy of the software is not

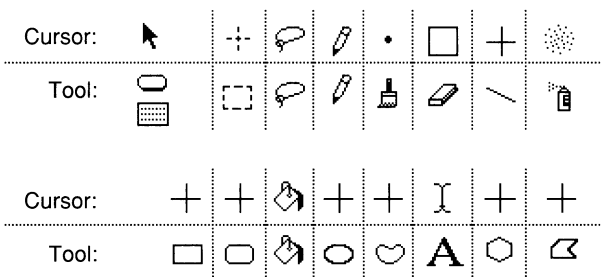
damaged or crippled. It simply means that your copy is currently set to a more restricted level of access. For details on how to set your *userLevel* (as this level is commonly called), see Chapter 25, "Cool Solutions to Uncool Problems."

If you choose any other tool but the browse tool from the tool palette, you are in authoring mode. This means that you do not intend to browse through the current stack as a user, but that you are going to change or add something to it. Typically, the only clear signal that you are in authoring mode is the shape of the cursor, which is often the plain pointer or one of the painting tools. Figure 2.2 shows some authoring and painting cursors with their corresponding tools.

Figure 2.2
The Tool palette, HyperCard cursors, and their tools.



The HyperCard Tool Pallete

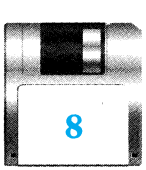


To choose the browse tool from any other tool and become a user or browser again (rather than an author), press command-tab from the keyboard.

If you want to look at authoring in HyperCard in more detail, there is a short tutorial on building stacks later in this book.

What Does HyperCard Consist Of?

Considering what we have learned about HyperCard's background and precursors, it is easy to define its constituent parts. HyperCard consists of a painting application, text editing and storing facilities, superior linking tools, and a programming language. All of these features may be used to create stacks. In order to read or create a stack you must in turn be running HyperCard. This can be a little confusing, especially if the stack "feels" like an application in its own right. In this case, you might think of the stack as an application that is using HyperCard as an engine to accomplish its task, like a crane using a truck's chassis and motor to move around.



In addition, HyperCard stacks have some built-in structure of their own. Each stack contains at least one *background*. A background is a layer of graphics and other objects that many screens may share. A good analogy is a printed form. When you fill out a pile of printed forms, the lines and boxes, as well as instructions on how to fill out the form,

are the background. The information that you enter on the form is in the foreground, or *card layer*. If someone else were to fill out the same forms, their information would be different from yours, but the form itself would be the same.

While a background may be shared by many screens, it is not necessary to have more than one card per background. Whereas a stack must contain at least one background, a background in turn must contain at least one card. You may choose to have multiple cards in a background, and you may also have multiple backgrounds in a stack. A stack that consists of only one background would probably hold a lot of similarly structured information, like an address book.

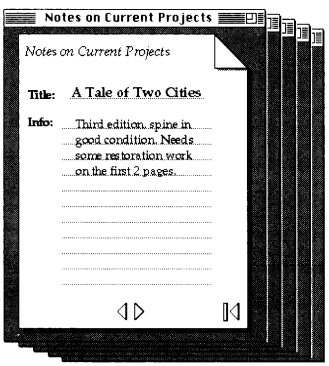

Notes on Current Projects
Notes on Current Projects
Title: A Tale of Two Cities
Info: Third edition, spine in good condition. Needs some restoration work on the first 2 pages.

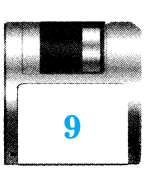
Figure 2.3
Similar information in one background.

Although you may not see more than one card in a stack at the same time, it is helpful to think of the HyperCard stacks in this way. As you navigate through a HyperCard stack, you typically move from card to card. If you were to move sequentially through a number of cards in a stack, when you reach the last card, continuing forward would bring you around to the first card of the stack again, not unlike a real-life Rolodex file.

Within this basic structure of backgrounds containing cards arranged in a Rolodex fashion, you can construct an incredibly varied number of creations.

What Does This All Mean?

Suppose you want your Mac to perform a task that no software company has ever tackled. You could try forcing some software that was designed for other work to do your job. For example, if you are moving your home or office, there are a hundred small tasks that might be better organized with your computer. But no one application is up to the peculiar needs of this task. You could use many separate applications and enter the same information into all of them, or you could force one application to fit. Or, you could learn to program the Mac in order to create a custom application, complete with a standard Macintosh graphical user interface, for a one-day event.



Therein lies the main problem with the Macintosh.

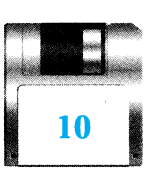
The Macintosh Toolbox and the Holy Brethren of the ROMs

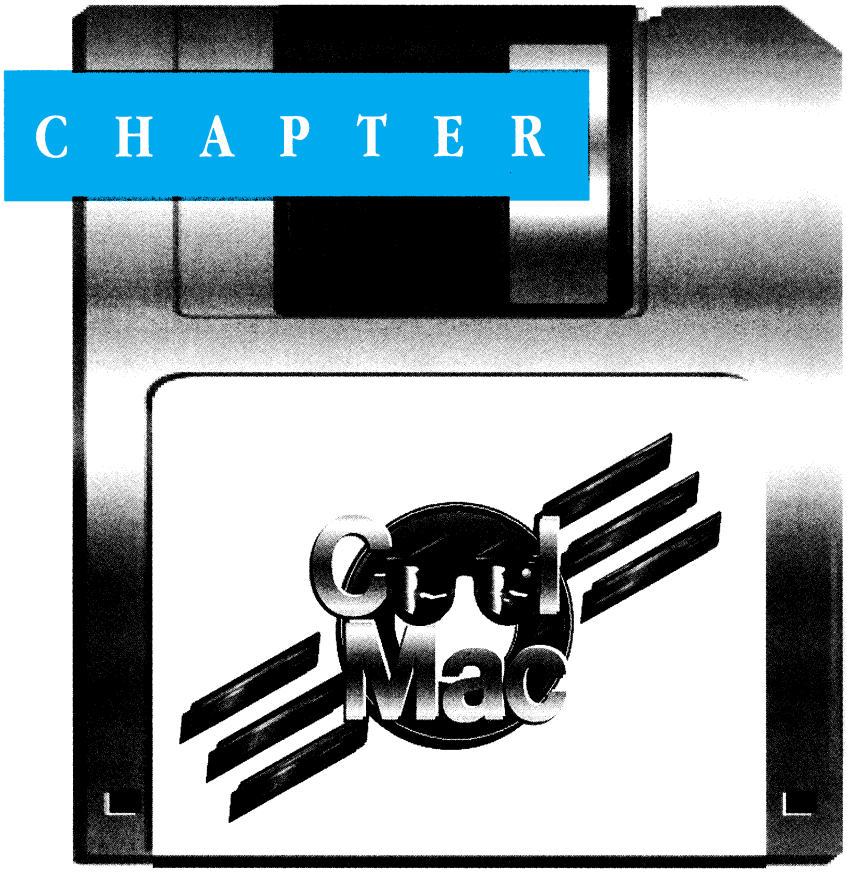
Learning to program the Macintosh is too difficult for most of us to master. In order to program the Mac you must become familiar with two large bodies of knowledge: (1) a standard computer programming language like C or Pascal, and (2) a mass of technical information concerning how to access the Macintosh Toolbox. What's the Toolbox? You might have heard the term batted around, often followed by other equally strange words like *ROMs, compile,* and *geek*. Maybe not the last one, but it probably came to mind. The Macintosh Toolbox is a set of special proprietary software routines burned into the silicon chips that are in every Macintosh. Because you can't change these routines, they are called Read Only, hence the term ROM or Read-Only Memory. The Macintosh ROM is responsible for many of the images that you see in the Macintosh user interface. The Chicago font is burned into the ROM. So are the pull-down menus, the parts of windows, the way that text wraps as you type it, the shape of the cursor arrow, and other things you take for granted in all Macintosh applications.

To learn all the details of conjuring the Toolbox, you must expend so much time and energy that you end up joining the club of Macintosh programmers: The Holy Brethren of the ROM. You must become intimately familiar with six volumes of a set of books called *Inside Macintosh,* the last volume of which could give you a back injury if you picked it up the wrong way.

What if you don't have the time or inclination to become a Macintosh programmer? Well, you could hire one to write, design, program, and debug your application for you at $100 or more an hour. If this is a small application to keep track of your dry cleaning, it hardly seems worth it, unless, of course, you are a dry cleaner.

HyperCard offers something that you just can't get in other software. HyperCard lets you develop custom solutions to problems that aren't easily solved by off-the-shelf software solutions. So, if you want your Macintosh to take on a specific job for you that no software company has ever dreamt of, and you don't want to enter into the mysteries of the ROM (perhaps becoming a techno-weenie in the process), the answer is HyperCard. No offense, Mac programmers.





3

HyperCard for the Browser

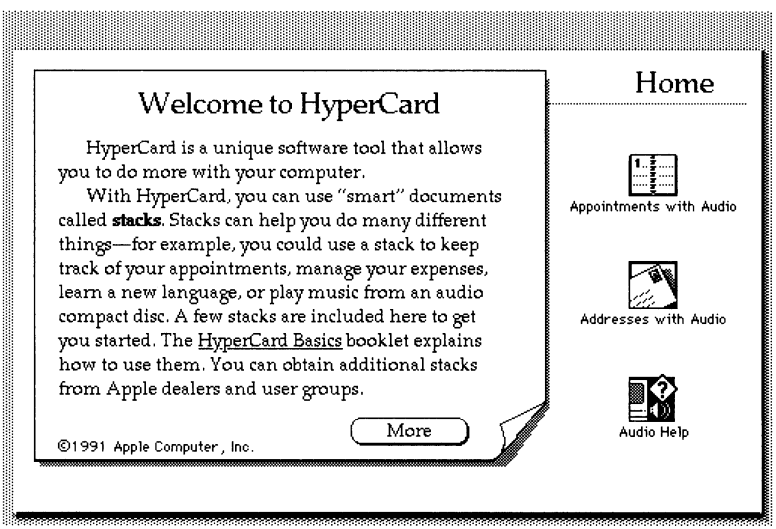
HyperCard is an electronic publishing medium. Someone creates a HyperCard stack and distributes it, and then you get to read it. Not too different from the world of print, is it? There are a few wrinkles, however, and this chapter helps you deal with starting up and entering your own information into a stack.

If you have used other Macintosh applications, much of what you know about how to use them still holds true. You click buttons with the mouse, drag down menus from the menu bar and pop-up menus on the screen, and type text into text fields.

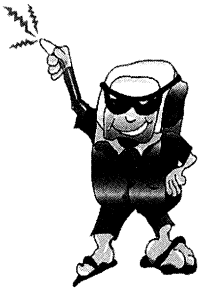
Starting Up HyperCard

Like nearly all Macintosh applications, you may start up HyperCard two ways: You can double-click the HyperCard application, or double-click a HyperCard document (stack) icon. Double-clicking HyperCard starts up the application and also opens a HyperCard stack called Home. You must have a Home stack to run HyperCard. Home is a special stack that HyperCard looks for as it starts up. If HyperCard doesn't see the Home stack, it will ask you to find it. If you can't find it, HyperCard quits right away. If you don't have a stack called Home, you can get around the situation temporarily by renaming a stack `Home` and placing it in the same folder as HyperCard. This is not a good regular practice, because most Home stacks take advantage of their special status to set several preferences as HyperCard starts up. If you received your copy of HyperCard with a System 7.0 Upgrade Kit, your Home stack looks like figure 3.1 when you start.

Figure 3.1
The first card of a HyperCard Home stack.


Welcome to HyperCard
HyperCard is a unique software tool that allows you to do more with your computer.
With HyperCard, you can use "smart" documents called **stacks**. Stacks can help you do many different things—for example, you could use a stack to keep track of your appointments, manage your expenses, learn a new language, or play music from an audio compact disc. A few stacks are included here to get you started. The HyperCard Basics booklet explains how to use them. You can obtain additional stacks from Apple dealers and user groups.
©1991 Apple Computer, Inc.
More
Home
Appointments with Audio
Addresses with Audio
Audio Help

The icons to the right of the screen, with the names **Appointments with Audio** and **Addresses with Audio**, are really buttons. This introduces one of the differences between HyperCard and the Finder desktop. In the Finder, one click selects an icon so that it can be opened, moved, duplicated, or deleted. To open an application or document in the Finder, you double-click the icon. In HyperCard, one click does not select an icon, but instead activates a script of the button to which the icon is attached. In the case of the **Appointments with Audio** and **Addresses with Audio** buttons, one click on each icon opens the stacks of the same name. Double-clicking is not encountered very often in HyperCard stacks, because a button that responds differently to single- and double-clicks requires the HyperCard author to write special HyperTalk scripts that carry out this behavior.



Opening Stacks from Within HyperCard

You can view other stacks by selecting **Open...** from the **File** menu, just as you would do in any other Macintosh application. You have the option of viewing the stack you are opening in a new window or closing the stack you are in as you open the new one. This last feature was added with version 2.0 when HyperCard gained the capability of opening more than one stack at a time.

Navigating Through a HyperCard Stack

As mentioned in the overview, each HyperCard stack must contain at least one card. This is the minimum that any stack must have in order to exist. If you wish to move through a stack with more than one card (most stacks do have more than one), there is a menu that is always available called the **Go** menu, shown in figure 3.2.

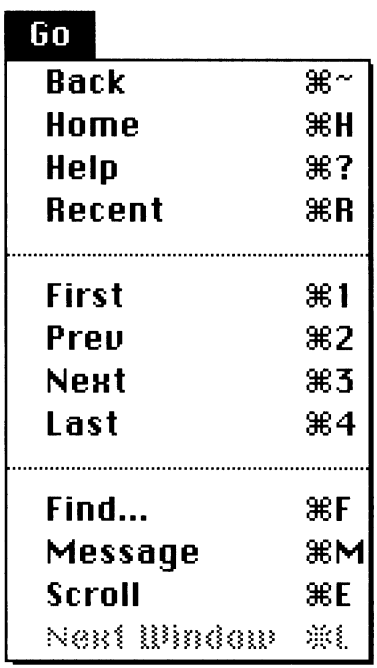


Figure 3.2
The **Go** menu.

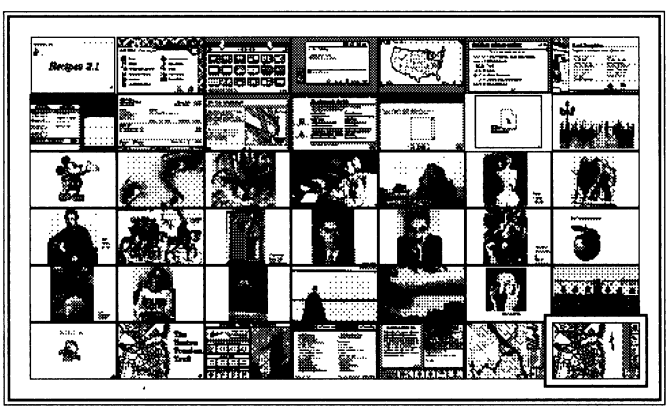
The **Go** menu includes navigation to the first and last cards of a stack, and also lets you move forward or backward through the stack.

HyperCard cards are created from the front of a stack to the back. When you create a new card in HyperCard, the new card is placed immediately after the card you were looking at. Cards can be referred to by number and by their assigned unique identification numbers as well. You can change the number of a card by cutting and pasting it elsewhere in a stack, but the identification number never changes.

The **Back** command returns you to the last card you were looking at.

Another menu item on the **Go** menu that can come in very handy while you are browsing is the **Recent** command. **Recent** shows, in miniature form, a picture of the last 42 cards you looked at, as shown in figure 3.3.

Figure 3.3
Images of cards seen recently shown by the **Recent** command.



These images are not just pictures; they also take you to the cards they portray. Just click a picture to go to the associated card. If none of the cards you wish to go to is shown in the recent window, click the current card—the picture that has a box around it.

You don't have to use the **Go** menu all the time, because it is the duty of the person who designed the stack to create buttons to navigate through the stack. All of the navigation commands in the **Go** menu can be performed by HyperTalk scripts. A HyperCard author who wants to allow you to easily navigate through a stack creates HyperCard buttons containing the scripts to take you forward, backward, and maybe to the first and last cards (if this is appropriate for their stack). A button to go to the next or previous card usually looks like a pointing arrow. Since most Americans and Western Europeans think of left as backward and right as forward, arrows pointing in these directions take you to the next or previous card in the stack. The bent arrow has come to mean go back, but don't count on this always being the case. A balloon, question mark, or light bulb means more information or help. Figure 3.4 shows some of these icons.

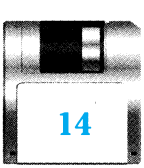


Figure 3.4
Some icons often used in HyperCard stacks.

Entering and Changing Text

You often see text in stacks. Text appears either as paint graphics or in containers for text, called *text fields*. If a text field may be added to or changed, your cursor changes from the browse cursor to an I-beam when you move the cursor over the text field. To type into a text field, click in the field with the I-beam and start typing. Your text appears where you clicked.

If you can't type in any text fields at all, this particular stack might be set to read only, or your level of access—or userLevel—might not be high enough to do so. See Chapter 25, "Cool Solutions to Uncool Problems," for more information about your userLevel.

Finding Text

The **Go** menu contains a menu item called **Find...**. When you select **Find...**, a special window appears. It is called the *message box,* and it is shown in figure 3.5.

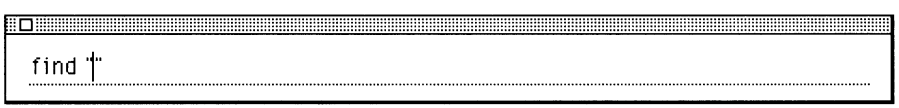
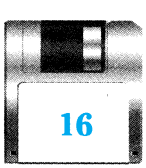


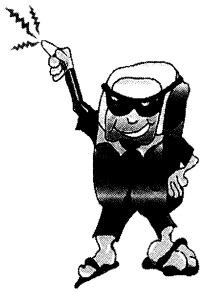
Figure 3.5
The message box ready to find text.

If you have not yet tried to find any text in the current session, your insertion point blinks between the quotes after the word *find*. If you tried to find text before, **Find...** remembers the last word you were looking for and places it between the quotes after the word *find*. In either case, type what you would like to find and press the return key. If your text is in the current stack, HyperCard immediately leaps to the first instance of it and you see a box around the found text. Press the return key again to go to another instance of the text you are looking for. If this particular instance is unique, you find this text in the same place again (the box reappears around the word). If there is no instance of the text in this stack, nothing happens.

Doing More than Browsing

You can use HyperCard simply as a reader or browser, but the real power of HyperCard is creating and customizing your own stacks or *stackware*. If you are interested in learning how to create and change stacks, the next chapter contains some useful information to help get you started.





CHAPTER

4

Creating HyperCard Stacks

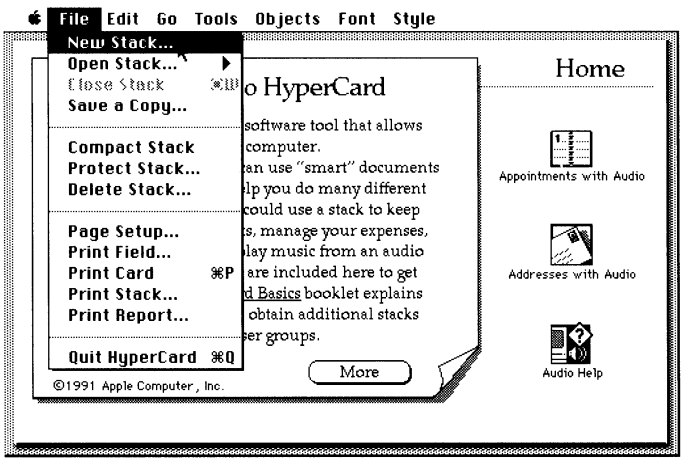
This is a short tutorial designed to take you quickly through the process of creating a new HyperCard stack. If you are interested in exploring the details of creating stacks, there are many books devoted exclusively to this subject. For now, here is a step-by-step introduction to authoring a simple stack. As you saw before, you are always viewing at least one HyperCard stack. If you do not specify a document when you start HyperCard, you always see the first card of your Home stack. This is as good a place as any to begin making a new stack.

Getting Started

Step 1: Choose **New Stack...**.

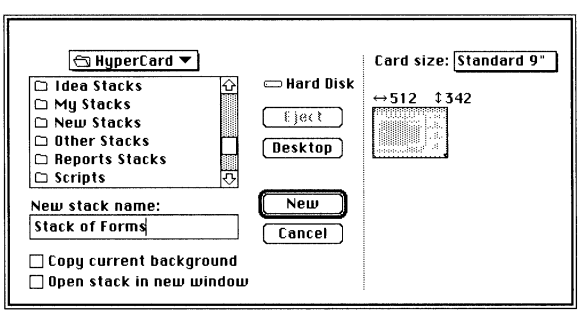
You create a new, empty HyperCard document by selecting **New Stack...** from the **File** menu shown in figure 4.1.

Figure 4.1
Selecting **New Stack...** from the **File** menu.



As figure 4.2 shows, you then see a standard save dialog box with a few added buttons and graphics.

Figure 4.2
The New Stack dialog box.



This new stack is designed like a form to fill in with name, address, and phone number. Before you create your stack, you must first name it. Name this stack `Stack of Forms`. HyperCard 2.0 added some extra options to this dialog box. You may specify a size for all the cards in your stack via a pop-up menu at the upper right with some standard size choices or a graphic that lets you set a custom size for the stack. If you end up setting a size that is wrong, don't worry, because you can always resize your stack later. Don't check

off the checkboxes **Open Stack In New Window** and **Copy Current Background** for now. You would check **Copy Current Background** if you intended to use the background you are in as a template for your new stack.

Step 2: Look at your new stack.

Click the **New** button. You see a blank white screen. This is the first card in your new stack. There is nothing here but the card and the background behind it, like a room ready to be decorated and furnished.

Step 3: Nose around a bit.

Choose **Stack Info...** from the **Objects** menu. You should see a dialog box with some information about your new stack, as shown in figure 4.3.

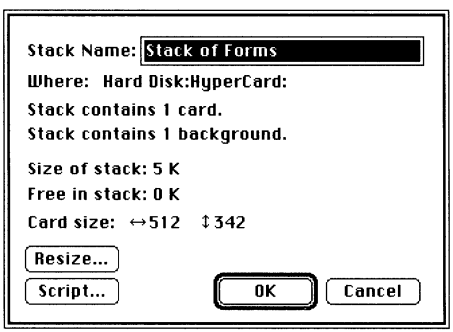


Figure 4.3
The Stack Info dialog box.

As you can see, **Stack of Forms** has one background and one card, the minimum required for a stack. Here you also can rename and resize your stack. Click **OK** to dismiss this dialog box.

Step 4: Make some background art.

To start the form, it is helpful to define areas on the screen which are pertinent to the form as well as areas for navigation buttons. You can use HyperCard's built-in paint tools to do this. To create graphics for your stack that show up on every card, you need to create them on the background layer.

The reason you would want graphics (or buttons and fields) to show up on a group of cards in your stack is so you don't have to create the identical layout for cards holding similar information. The IRS doesn't type up a brand-new tax form for every taxpayer to fill out (although we might wish they had to), they use the same text for every form. The information changes, but the form—the background—doesn't change.

To start editing the background, rather than the card, choose **Background** from the **Edit** menu. Little will change except for the menu bar, which should have zebra stripes or hatch marks along it, like those in figure 4.4.

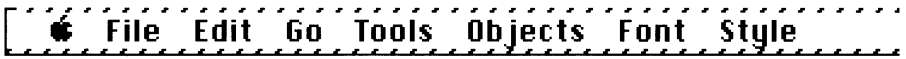


Figure 4.4
Hatch marks indicate that you are editing the background layer.

These marks are the only clue that you are editing the background domain. If there were card-specific information, like a card graphic or text for this card, it would be easier to see the shift to the background layer, because this information would disappear.

Because you are going to use the paint tools of HyperCard, you can make them more available by tearing the **Tools** palette off the menu and dragging it to a convenient place, as shown by figure 4.5.

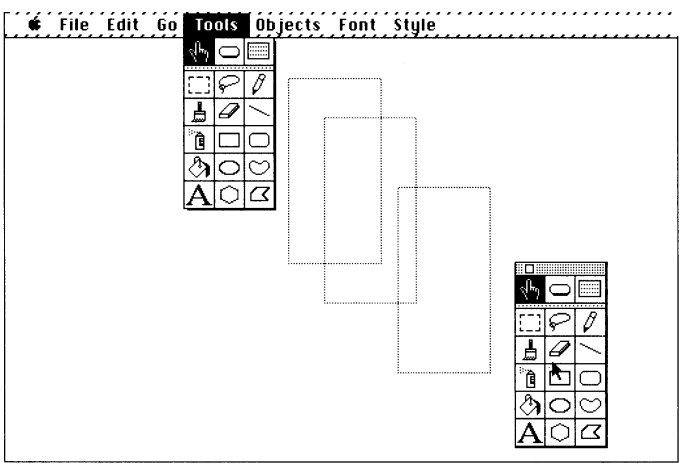


Figure 4.5
Tearing off the **Tools** menu to become a tool palette.

Let's create a rectangle to enclose the content area of the form. To do this, choose the rectangle paint tool from the **Tools** menu or palette. Drag a new rectangle graphic on the background, using about two thirds of the screen's area. Create a second, smaller rectangle below the first for the navigation area. Your background should look like figure 4.6.

Finally, add some gray background paint to the areas not included in the form or navigation areas. Do this by using the paint bucket. Choose the paint bucket and then choose the gray pattern from the **Patterns** menu. The **Patterns** menu can also be torn off to become a palette, as shown in figure 4.7.

Click once with the paint bucket cursor somewhere on the background, but outside of the two rectangles you made earlier. You should see something similar to figure 4.8.

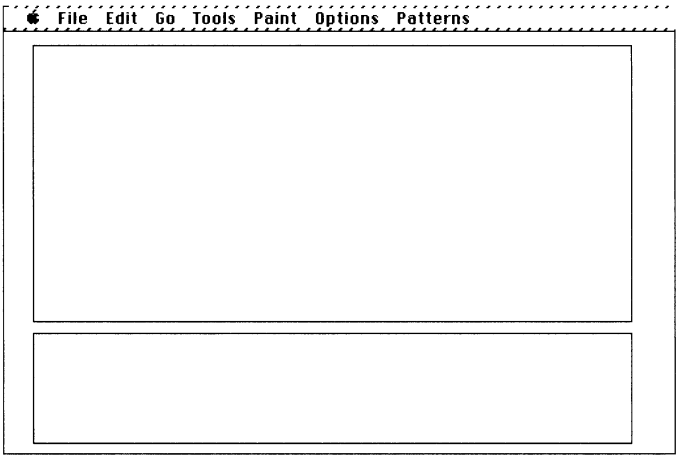


Figure 4.6
The background after drawing two rectangles.

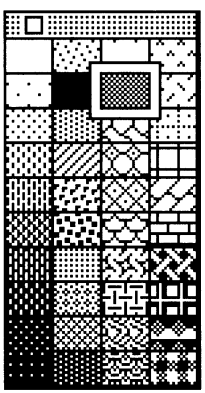


Figure 4.7
The pattern palette.

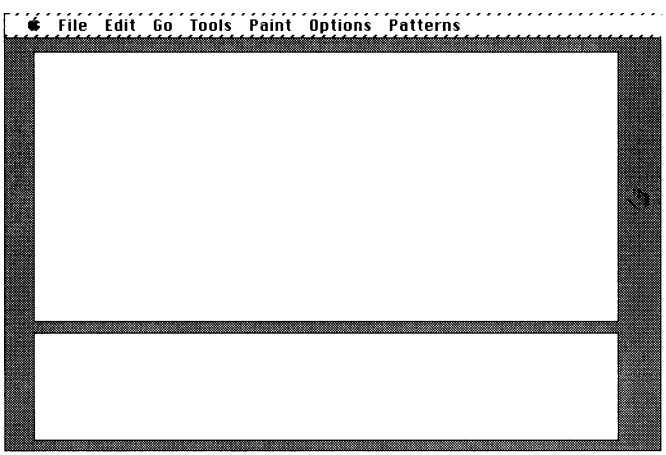


Figure 4.8
The finished background graphic layout for this background.

Step 5: Make some background fields.

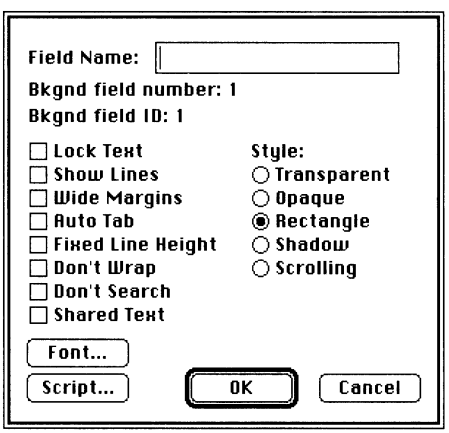
Now that there is some sense of a layout for the information, you can create some containers for text and navigation tools. To create a new background field, choose **New Field** from the **Objects** menu.

A new text field appears. Also, the field tool on the tool palette is selected. Move your new field into position by clicking and dragging somewhere in the center of it. Resize it by clicking and dragging near an edge or corner. Double-click the field to see the Field Info dialog box, as shown in figure 4.9.

Name this field by typing `Name and Address` in the box. Also, change the style of the field by clicking the radio button called **Rectangle**. Click **OK** to confirm your changes.

Choose **New Field** from the **Objects** menu; reposition and resize it as well. Rename it Phone Number and change its style to **Rectangle**. Your background should now look something like figure 4.10.

Figure 4.9
The Field Info dialog box.



Step 6: Make some background buttons.

You create new background buttons in the same way you created new fields. Here you can create buttons to navigate forward and backward through the stack. Although these navigation commands are available from the **Go** menu, it might be more convenient to include them on the screen as buttons. To add a new button to the background, choose **New Button** from the **Objects** menu. A new button appears and the button tool on the tool palette is selected. Double-click the new button to see the Button Info dialog box, shown in figure 4.11.

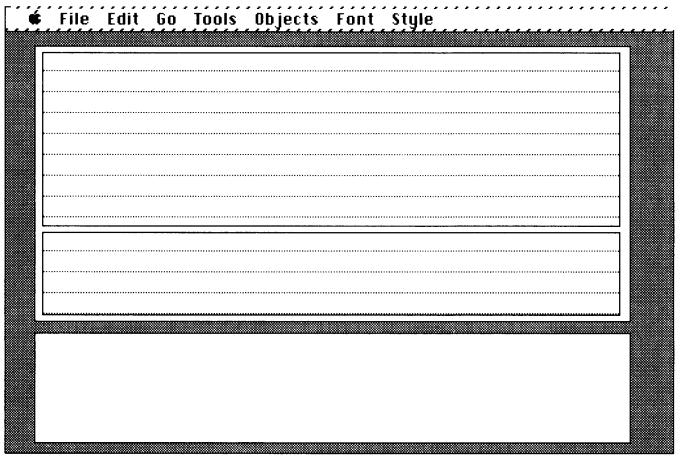


Figure 4.10
The background with new fields.

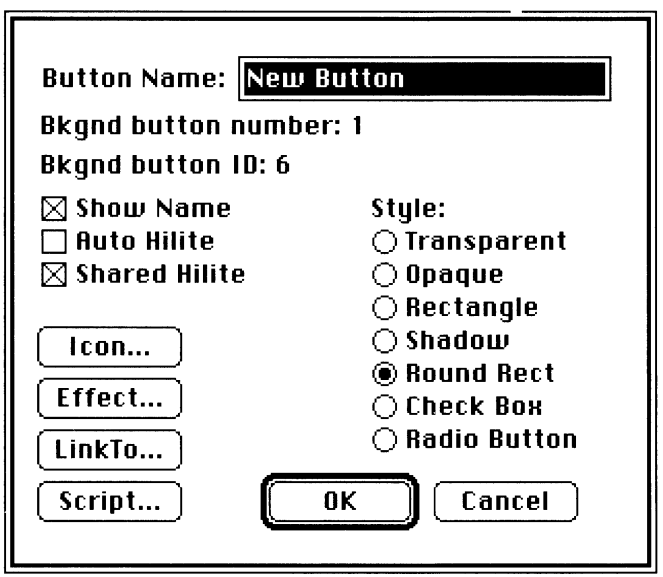
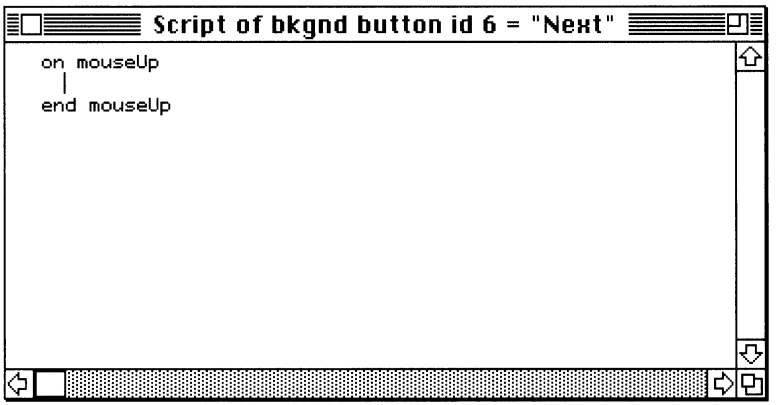


Figure 4.11
The Button Info dialog box.

Name this button `Next`. To give this button the capability of navigating forward through this stack, the button must have an attached HyperTalk script, the programming language of HyperCard. Click the **Script...** button. A script-editing window appears, as shown in figure 4.12.

Figure 4.12
A script-editing window.



The insertion point is blinking on the second line. Type in the words

```
Go to next card
```

Close the window and click **Yes** to save your changes.

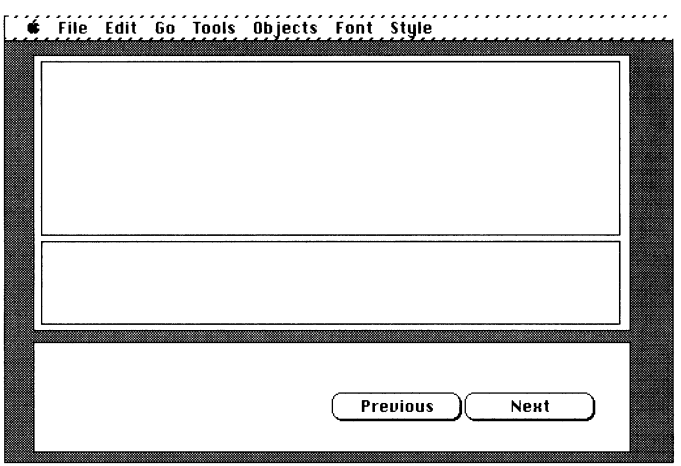
Reposition your button so it is within the lower rectangle.

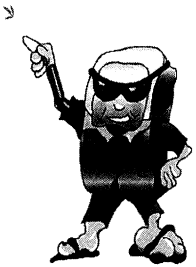
To create a second button to go to the previous card, again choose **New Button** from the **Objects** menu. A new button appears. Double-click this new button to see the Button Info dialog box that you saw in figure 4.11. Name this button **Previous**. Click on the **Script...** button. You see a text-editing window like the one in figure 4.12. This time, type the words

```
Go to previous card
```

Reposition this button so it is within the lower rectangle. Your screen should look like figure 4.13.

Figure 4.13
The finished stack.





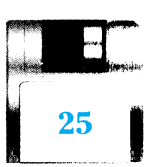
Using Your New Stack

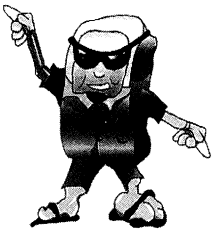
Now that you have built this stack, you are ready to use it. Choose **Background** again from the **Edit** menu. This brings you back to the card layer. The last step is to stop being an author and become a user again. Do this by choosing the browse tool.

You are now browsing through your own stack. Enter text into the fields. If you would like to make a new card for this stack, do so by choosing **New Card** from the **Edit** menu. Clicking on the **Next** and **Previous** buttons you made takes you from card to card. You can tell that you are navigating by changes in card-specific information, like the text in the fields. If you would like to add a card field, you can do it the same way you added the background fields, except you must be operating in the card layer (no hatch marks on the menu). You can think of a card field as a Post-it note for a specific form, whereas the background fields are standard places for information to appear in the form. Remember that if you want to type text in either a background or card field, you need to be a user, using the browse tool.

Moving On...

In this tutorial, you have created a simple stack. It contains graphics, text, and buttons with HyperTalk scripts in them. Many HyperCard stacks are elaborations on this basic structure. Developing HyperCard stacks is often a gradual, additive process. One other nice thing about HyperCard is that you rarely have to start from scratch, as we did in this tutorial. If you like the structure or graphics of a stack, you may save a copy of the stack you are looking at by choosing **Save a Copy...** from the **File** menu. After you have a copy, delete all the cards that you do not want, and then empty out the fields where you want to put your own information. Then customize your new copy by adding features you want to see in it. HyperCard lets you build features on straightforward beginnings or the starting points created by others. The first stack of the *Cool Mac Stacks* collection, Recipes, closely follows the model of the stack in this tutorial, but adds much more. To see how it works, read on.





5

About External Commands

Several stacks included in the disk contain special additions to HyperCard called External Commands, or XCMDs (pronounced "ex–commands"). This chapter helps explain what XCMDs are and how they can add new capabilities to HyperCard.

What's This X-Thing?

In order to understand what XCMDs are, it's helpful to understand the special structure of all Macintosh documents and applications. Every Macintosh file has two parts or pockets of information called *forks*. Files may have information in both a *data fork* and

a *resource fork*. The data fork of a document contains the words that you read in a text file, the numbers of a spreadsheet, and the objects or dots of a graphics program. The resource fork is used more frequently by applications. It contains the application's unique icon, menu names, fonts, patterns, and color tables. More importantly, the resource fork contains the instructions to do the job of painting, or calculating, or whatever that application does, called CODE resources. However, documents can also contain information in their resource forks. HyperCard stacks often contain icons, sounds and special kinds of programming code called XCMDs in their resource fork.

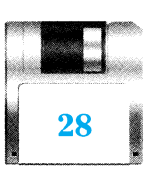
What XCMDs Can Do for You

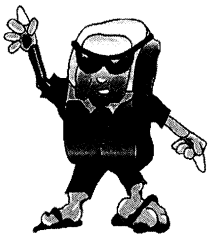
So an XCMD is a piece of programming code. What does that get you? Plenty! An XCMD typically adds a new word to Hypertalk, the scripting language of HyperCard, while you are looking at the stack where the XCMD lives. An XCMD is then called to action by a HyperCard script, and once it is running, it can do all sorts of amazing things. XCMDs can control videodisk players and CD ROM drives. XCMDs can let HyperCard access the Macintalk speech driver, giving your stacks synthesized speech. XCMDs can fill in places where HyperCard doesn't support the full Macintosh interface, giving you pop-up menus or lists in dialog boxes. XCMDs can play animations from your hard disk, compressed sound, and, eventually, QuickTime movies (compressed video). Actually, XCMDs encompass two kinds of code: XCMDs (e*X*ternal *CoM*man*Ds*) and XFCNs (e*X*ternal *F*un*C*tio*Ns*). XCMDs actually do something, such as speak, play, or flash; XFCNs tell you something, such as the name of the chosen printer or how many colors the monitor is set to display. XFCNs return a value back to HyperCard that you can use in your script or record in a field.

How to Get XCMDs to Do Their Stuff

XCMDs and XFCNs, as mentioned before, live in the resource fork of a HyperCard stack. They are called from a HyperCard script, and often the script communicates some information to them, called *parameters* or *arguments*. There is an XCMD called flash that comes as an example with HyperCard. You control the flash XCMD by telling it how many times to flash the screen. So if you type the following script into a button:

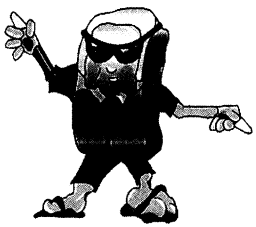
```
on mouseUp
    flash 3
end mouseUp
```





HyperCard calls the flash XCMD and the parameter which makes the XCMD flash the screen three times. Flash is the only XCMD that actually lives in the HyperCard application. XCMDs that you see on the example disk are in the resource fork of each stack that demonstrates or uses them.

As you might have guessed by now, to get XCMDs into your stacks, it is necessary to have a tool for copying and pasting information from one file's resource fork to another. The best-known and most widely distributed application for doing this is called ResEdit. ResEdit, at the time of this publication, is in version 2.1.1, and is readily available from user groups and on-line services (see Chapter 26, "On-Line Services and User Groups," for more information about contacting these sources). ResEdit is a powerful tool. Because it gives you direct access to the very guts of applications and documents, use ResEdit with care. In fact, use it only on copies, not your original files.



CHAPTER

Stack Name:	Recipes
Folder	Information Enhancement
Size:	64K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Beginner
Special Requirements:	None

6

Recipes: A Tasty Database

You might have heard that an electronic recipe file is a good reason to have a home computer, along with family finances and checkbooks. However, the real problem most people have with entering their personal information into their computers is that taking time to enter all the data isn't really worth it. Yes, you do get rid of some messy index cards, and you can show off to your friends, but until you can program your dinner into the replicator unit in the wall like Mr. Spock or Jane Jetson can, recipes typed into your computer might not be as useful as paper—unless the computer offers something that you can't get on paper. The Recipes stack proves that it pays to get recipes into your Mac.

Recipes is a stack for holding recipes. The stack is closer to Apple's original examples of what you can do with HyperCard than anything else in this collection. Recipes are the kind of card-based information that HyperCard stacks were made to hold. However, there are some bells and whistles thrown in that make this stack a bit more than just a simple electronic card file.

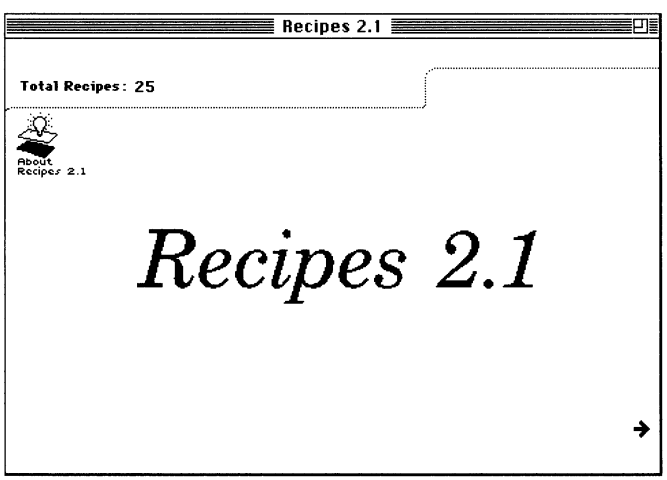
How to Use This Stack

Recipes is one of the stacks in this collection that adds its own menu to the menu bar, shown in figure 6.1.

Figure 6.1
The **Recipes** Menu

This feature was added with HyperCard 2.0 and helps to make the stack's interface more like a standard Macintosh application. The **Recipes** menu includes commands to sort recipes alphabetically by title, to import and export recipes, and to show a list of all the recipes, here called an index. Other menu items are discussed later in more detail. See figure 6.2 for the cover card of the Recipes stack.

Figure 6.2
The Recipes cover card.



To browse through the recipes, click the arrows at the bottom. You flip from the top, or cover, card to the first recipe, and then to the next. Clicking on the far left arrow pointing to a vertical line takes you back to the cover card again. Figure 6.3 shows a typical recipe card with these arrows labeled.

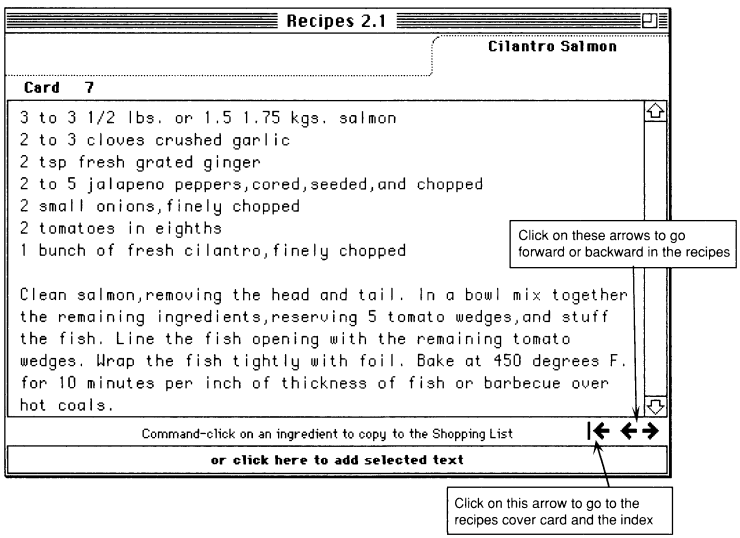

Recipes 2.1
Cilantro Salmon
Card 7
3 to 3 1/2 lbs. or 1.5 1.75 kgs. salmon
2 to 3 cloves crushed garlic
2 tsp fresh grated ginger
2 to 5 jalapeno peppers,cored,seeded,and chopped
2 small onions,finely chopped
2 tomatoes in eighths
1 bunch of fresh cilantro,finely chopped
Clean salmon,removing the head and tail. In a bowl mix together the remaining ingredients,reserving 5 tomato wedges,and stuff the fish. Line the fish opening with the remaining tomato wedges. Wrap the fish tightly with foil. Bake at 450 degrees F. for 10 minutes per inch of thickness of fish or barbecue over hot coals.
Click on these arrows to go forward or backward in the recipes
Command-click on an ingredient to copy to the Shopping List
or click here to add selected text
Click on this arrow to go to the recipes cover card and the index

Figure 6.3
A typical recipe card.

To see a list of the titles of all the recipes in the stack, go to the cover card and select **Show Index** from the **Recipes** menu, or press command-S. You see a scrolling field appear, as shown in figure 6.4.

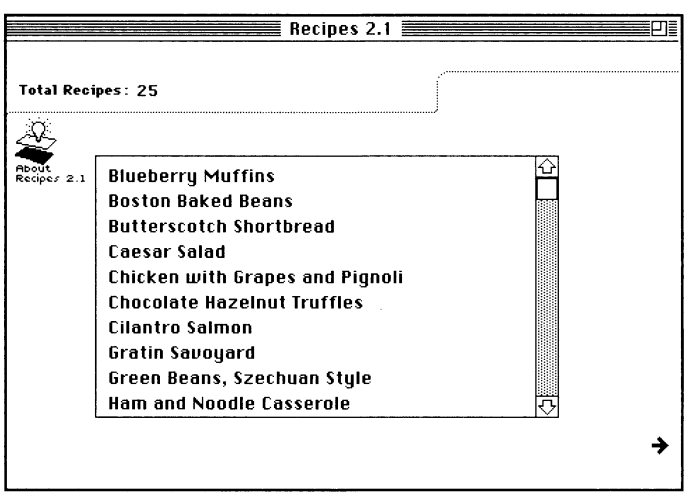


Figure 6.4
The Recipes index on the cover card.

If you click any line of text in the list of recipes, you jump to the corresponding recipe. In this stack, the recipe index is updated whenever you sort the stack using the **Sort Recipes** menu command, so if you want any new recipes that you add to appear in this list, you must sort the stack first. Fortunately, this doesn't take very long, unless you collect more than two or three hundred recipes.

The Shopping List and Search Features

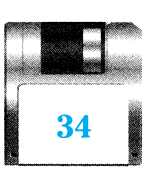
One of the advantages of having recipes in HyperCard is that you can create a shopping list of ingredients you need for a recipe while reading the recipe. The computer copies the ingredients for you. To add an ingredient to the shopping list, either click the word (such as *sugar*) while pressing the command key, or select some text and click the long rectangular button at the bottom labeled **or click here to add selected text**. To see what you have accumulated on the shopping list so far, choose **Shopping List** from the **Recipes** menu or press command-L. You can print the shopping list at any time.

Getting Recipes into and out of This Stack

You can add recipes of your own by selecting **New Card** from the **File** menu and typing a title and recipe into the appropriate fields. If you have a plain text recipe on your disk, you can import the text recipe into the stack by choosing **Import Recipe...** from the **Recipes** menu. **Export Recipe...** saves the contents of a recipe card to a plain text file. This is convenient if you want to send a recipe to a friend or include it in a letter (although copy and paste also work from HyperCard to other applications like word processing). If you need a hard copy of the recipe (assuming your Mac is not in the kitchen), choose **Print Recipe** from the **Recipes** Menu. If you can't see all of the recipe in the scrolling field on the card, don't worry. **Print Recipe** prints all the text of the recipe.

La Pièce de Resistance...

Another advantage of having your recipes on the computer is a search feature. Let's say that you have chicken and onions in the house. Wouldn't it be nice to see what you could make with these ingredients? To do this, choose **New Search...** from the **Recipes** menu. You see a dialog box asking what you would like to search for. You can search for just one ingredient, such as `chicken`, or you can search for more than one. If you type more than one word here, such as `chicken` and `onions`, HyperCard would search for recipes containing both words. After you type what you would like to search for, you see a progress monitor, and then a list of matching recipes, as shown in figure 6.5.



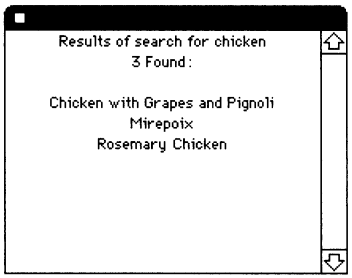
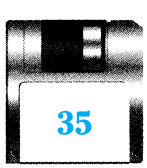
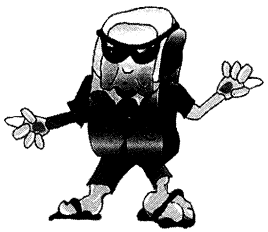


Figure 6.5
A list of matching recipes using `chicken`.

If you click any of the matching recipe titles in this list, you jump to the recipe!

A batch search feature is the kind of task all databases perform. They give you a list of "hits" to a search. If you choose **Find** from the **Go** menu and then type a word in the message box, HyperCard finds the first instance of that word and takes you there. The search script in this stack does the same thing, but continues to go forward, looking for other entries, until HyperCard arrives at the first card again. While the script locates instances where the word is found, it takes note of where those instances occurred. When HyperCard comes full circle, it puts the list of found cards into a field where you can see them. You can easily adapt the batch search in this stack to another stack. If you had a stack to keep track of your favorite movies, instead of searching for broccoli, you could search for Harrison Ford. It might be worth exploring. Dig in.





CHAPTER

Stack Name:	**Phone->Acronym**
Folder	Information Enhancement
Size:	16K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Beginner
Special Requirements:	None

7

Phone->Acronym: When Numbers Are Not Enough

You've probably seen advertisements that give you a phrase to dial instead of a telephone number, like BIG-DEAL, or 1-FOR-YOU. For a small stack, Phone->Acronym packs a lot of processing power. It generates all the possible combinations of letter equivalents that you can derive from the telephone dial (or keypad) for a telephone number, so that you can determine if your phone number contains any words that you can use.

How to Use This Stack

Using Phone->Acronym is very easy. The stack consists of one main screen, shown in figure 7.1.

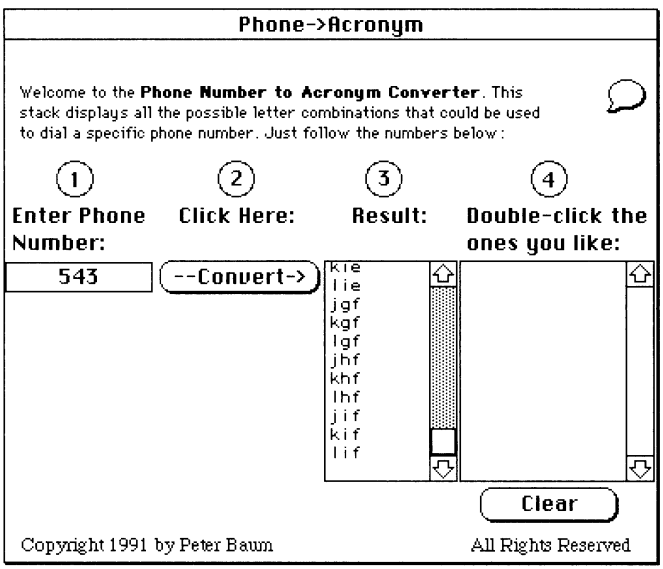


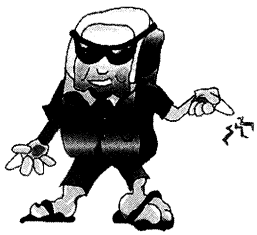
Figure 7.1
The Phone->Acronym main card.

Generating useful acronyms is a three-step process.

1. First, fill in a phone number you would like to convert. Unfortunately, as you may have noticed, the numbers 1 and 0 don't have associated letters on the phone dial, so you must leave those out. Fortunately, some shorter words mixed with numbers also work well (like 126-1243, which is 1-TO-1-TIE, or 228-2646, which is 22-TANGO). In addition, make sure that you use only the numbers of your phone number—not dashes or parentheses.

2. Click the **--Convert->** button. The stack then churns away generating all the possible combinations. The amount of time it takes to calculate the results is directly proportional to how many numbers you enter in the field. If you enter seven numbers, this can take quite a while.

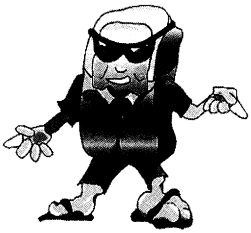
3. After the stack is done, the results show up in the scrolling field in the middle. This can include many combinations. Most of them are nonsense, like *hdjr* and *idkr*, but once in a while you see some real words, like *help* and *gels* (from the same numbers). If you see any combinations that make sense, double-click them. They are then copied to a holding field at the right for you to use later. Maybe you'll get lucky!



Computers as Coincidence Generators

Computers can help out in many calculation-intensive word and number games. Years ago, a programmer named Michael Morton brought a program called ARS MAGNA to the Mac. Originally conceived on mainframe computers, ARS MAGNA created anagrams. An *anagram* is a word or phrase created from the reordered letters of another word (as ARS MAGNA is of ANAGRAMS). Michael Morton's ARS MAGNA churned out all the possible combinations of words from another. He found that his name produced *Mr. Machine Tool*. I learned, much to my satisfaction, that my full name, David Landes Drucker, produced *Riddled Cadaver Sunk*, and *Cruel David's Dark Den*, which I placed on the door to my office. I gave friends anagrams of their names as presents.

Applications like ARS MAGNA and Phone->Acronym might not be particularly practical, but they put the dull-witted workhorse of computer chips to use as a generator of interesting possibilities. We can pick through the heaps of gibberish to find the useful coincidences. Let your Mac help you find useful and curious coincidences in your life, starting with your telephone number.



CHAPTER

Stack Name:	**WildAbacus**
Folder:	Information Enhancement
Size:	34K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Beginner
Special Requirements:	UserLevel must be set to 3 or above.

8

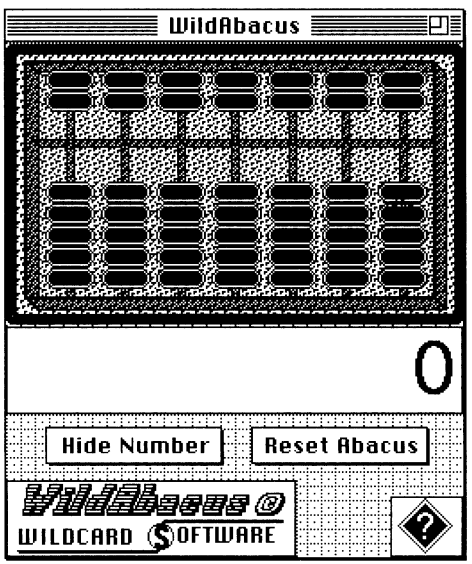
WildAbacus

Do you know how to use an abacus? If the answer is no, don't be ashamed. For most Westerners, this ancient calculator is not a household word. The abacus was first invented by the Greeks. The name, in fact, derives from the Greek word *abax*, meaning "board" or "calculating table." The Greeks and Romans used pebbles or metal disks as counters, because their system of writing numbers did not make it easy to "carry ten" the way we do when we solve arithmetic problems on paper. The Roman numeral system, in which letters represent numbers, was dominant in Europe for nearly 2,000 years. However, even simple addition of Roman numerals—XIII + LXIX, for example—was difficult. The abacus is still used by shopkeepers in the Orient, Russia, and the Middle East, but in our world of electronic calculators (or calculator desk accessories), they are somewhat of a mystery. The WildAbacus stack helps reacquaint us with this mathematical artifact.

How to Use This Stack

This HyperCard abacus is a simulation of a more modern abacus—the kind that uses a bar in the middle. Figure 8.1 shows the abacus screen.

Figure 8.1
The WildAbacus card.



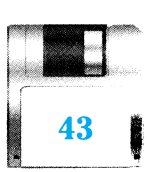
To start using the abacus (which I suppose you would have to resort to if your calculator's batteries ran down), click the **In an Emergency, Break Glass** button. This just sets up the abacus for your calculation.

The bar separates two groups of beads. Each bead on the bottom stands for a one, and each bead on the top stands for a five. Each rung, from right to left, increases by a factor of ten. Numbers are added by moving the beads toward the bar and subtracted by moving them away from the bar. To move a bead or set of beads, click them. This abacus also displays the answer in arabic numbers, so you can see quickly the results of a move. After a while, you can hide the arabic numbers by clicking the **Hide number** button. This can test you to see if you are reading the abacus properly. You'll probably get the hang of it very quickly.

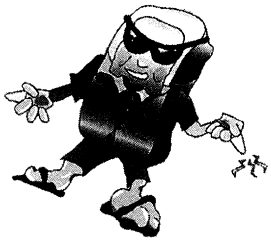
HyperCard as Simulation Software

With stackware, you can explore the details of a machine or concept by experiencing it instead of just reading about it. This makes HyperCard an important educational tool.

Museums often set up computers with simulations near exhibits to give visitors a more vivid appreciation of the subject. Stackware has been used to demonstrate everything from the flow of electrons to the rotation of planets. WildAbacus gives you a chance to experience an abacus without having to go out and get your hands on one (they can be hard to find these days). HyperCard simulations of other software are often used by developers to give users a guided tour of their applications. You might have taken such a tour without ever knowing it was a HyperCard stack.



CHAPTER



Stack Name:	**Sunrise Demo**
Folder:	Information Enhancement
Size:	56K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.0
Level of Expertise:	Beginner
Special Requirements:	None

9

What Time Is Tomorrow's Sunrise?

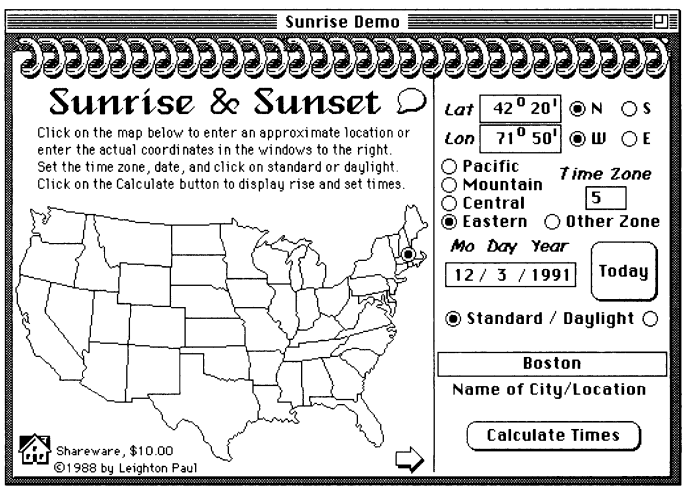
This stack has a special capability. If you give it some information about where you are, the time zone, and a date, the stack calculates the times sunrise and twilight occur, down to the minute. This stack demonstrates a special external calculation function and computes the sunrise times for anyone in the continental United States.

How to Use This Stack

To use the Sunrise Demo stack:

1. Open the stack. You see a map of the continental United States and some fields for information at the right. Figure 9.1 shows the stack.

Figure 9.1
The Sunrise Demo stack.


Sunrise Demo
Sunrise & Sunset
Click on the map below to enter an approximate location or enter the actual coordinates in the windows to the right.
Set the time zone, date, and click on standard or daylight.
Click on the Calculate button to display rise and set times.
Lat 42° 20' N S
Lon 71° 50' W E
Pacific
Mountain
Central
Eastern
Time Zone
5
Other Zone
Mo Day Year
12 / 3 / 1991
Today
Standard / Daylight
Boston
Name of City/Location
Calculate Times
Shareware, $10.00
©1988 by Leighton Paul

If you know the longitude and latitude of the place where you live, enter them in the fields at the right.

If you have ever located your city or town in the Map Control Panel, then you can find your longitude and latitude there.

If not, don't despair. This stack has a very clever means of determining your longitude and latitude. Click somewhere on the map of the continental United States. You hear a click, and a radio button jumps to the place where you clicked. The latitude and longitude for the location you chose appear in the fields at the right. This method might not be as accurate as finding your location in a geographical atlas, but it's probably close enough for a decent sunrise calculation.

2. Click the radio buttons at the right to specify your time zone and whether you are on Daylight Savings Time or Standard Time. If you would like to see the sunrise time for today, click the **Today** button. Otherwise, enter a date for which you would like to see the sunrise. I usually pick tomorrow's date.

3. Fill in the name of the city or location you would like to see when the values are calculated and formatted.

4. Click the button on the bottom called **Calculate Times**. You should see something like figure 9.2.

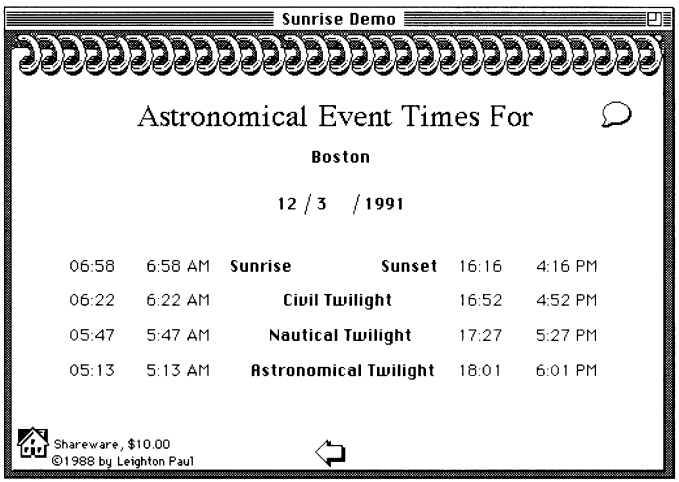

Sunrise Demo
Astronomical Event Times For
Boston
12 / 3 / 1991
06:58 6:58 AM Sunrise Sunset 16:16 4:16 PM
06:22 6:22 AM Civil Twilight 16:52 4:52 PM
05:47 5:47 AM Nautical Twilight 17:27 5:27 PM
05:13 5:13 AM Astronomical Twilight 18:01 6:01 PM
Shareware, $10.00
©1988 by Leighton Paul

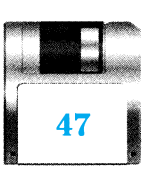
Figure 9.2
The results of a sunrise calculation.

Now I can complain that I really do get up before the crack of dawn, at least in December.

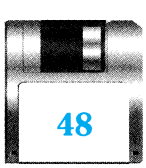
Notice that this stack is a demonstration. It incorporates a subset of a set of external functions that calculate astronomical events, including the times and dates for sunrise, sunset, the phases of the moon, high and low tides, and so on.

HyperCard as a Front End

This stack is an example of a popular concept known as a *front end*. A front end is a case of a computer program acting as a friendly face between you and a difficult task or calculation. For example, when you go to the bank, the automatic teller machine acts as a front end between you and the sophisticated and complex machines of the bank. You carry out a dialog with an easy-to-use local screen, and your transaction is communicated to the bank's big computer, without you having to deal with it directly. The Sunrise Demo stack is a front end between you and the code that calculates the sunrise times. HyperCard provides an attractive screen for getting the information into the calculation code and also delivers the information in a readable format. HyperCard is often used as a front end to bigger computers, often by network cables or even by using a telephone line. The person using the Macintosh sees a handsome screen with all of the standard Macintosh icons, text boxes, and pull-down menus, but the mainframe or minicomputer sees a person with a terminal dialing up.



There are more and more examples of front ends using the Macintosh; you might use some of them and not even know it! The on-line service America Online uses special software to front end its computer. On your Mac, you see pretty color icons, windows, menus, and lists. The host computer in Virginia receives bursts of control characters and streams of data from your modem, and responds with streams of data. The America Online software then translates the host's responses into meaningful Macintosh events, like opening a window and displaying a list. Someday, we'll all be able to build (or buy) front ends to many difficult tasks. I'd like a front end on my Mac to program my VCR. Then I wouldn't get a four-hour recording of the Weather Channel.



CHAPTER

Stack Name:	**Renaissance Man**
Folder:	Fun and Games
Size:	94K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Beginner
Special Requirements:	None

10

Renaissance Man: A Word Game with a Twist

When I was really bored in school, I used to play Hangman with my friends in the back of the class. It's a simple guessing game. Your opponent thinks of a word and then draws lines in a row on a piece of paper representing each letter of the word. You then guess what letters, one at a time, are in the word. If you correctly guess a letter in the word, your opponent writes it in the correct space (or spaces) where it appears in the word. If the letter you guess is not in the word at all, your opponent starts to draw a picture of a hanging man (usually a stick figure on a simple gallows), one body part at a time. If you guess the word before the man on the gallows is

completed, you win. If not, your man dies and your opponent wins. The Renaissance Man stack plays Hangman with you. HyperCard picks the word, and you try to guess it. The difference with this version of the game is that instead of a hanging man, something else appears.

How to Use This Stack

When you open the Renaissance Man stack, you first see a cover card, shown in figure 10.1.

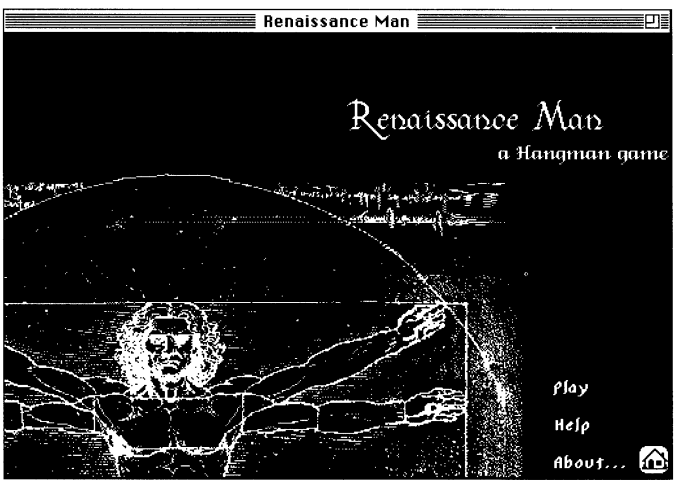


Figure 10.1
The Renaissance Man cover card.

The cover card takes you back to Home, Help, or the game card or screen shown in figure 10.2.

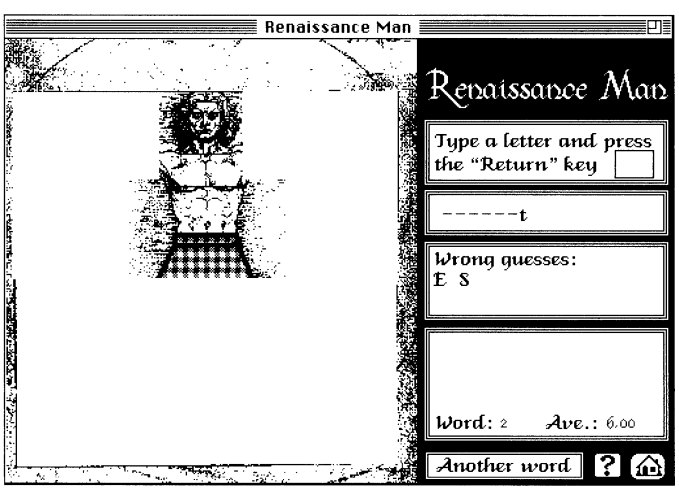
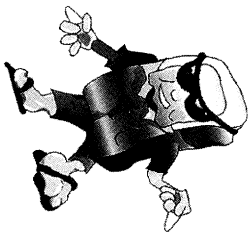


Figure 10.2
The Renaissance Man game screen.



Renaissance Man chooses a word at random and represents the number of letters as a series of horizontal lines.

To play the game, type a letter you think might be in the word. Your letter appears in the box at the top. Press return.

The **Rman** menu lets you change several settings of the game, including the degree of nudity in the picture—either **Modest**, with boxer shorts, or **Unashamed**, in which everything from the original is shown. The **Rman** menu also has a command to turn on or off the sound. This includes sounds for correct and incorrect guesses, as well as some little tunes when you win or lose the game. From the **Rman** menu, the number of guesses it takes to completely reveal the picture can be set to either six or eight.

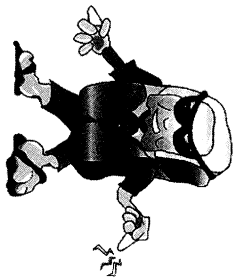
If your letter was in the word, you see your letter appear in one or more spaces at the top. If not, you see parts of a picture appear at the left. If you haven't figured it out by now, the picture is the famous illustration of the proportions of the human body, done by Leonardo da Vinci, a real Renaissance man.

To start another game, click the button **Another word** at the bottom of the card.

Renaissance Man has a list of 1,000 words from which to choose, so you probably won't see the same words very often. If you would like to personalize Renaissance Man to include the proper names of your children or pets, you can create your own word list. Do this by choosing **Your Own Word List** from the **Rman** menu.

HyperCard Can Play Games with You

Renaissance Man is a real game in HyperCard, complete with synchronized graphics and sound. The graphics are well done, and the game's large vocabulary shows HyperCard's unique blend of application and document. The game has the vocabulary, because the vocabulary is in the stack. Have fun!



CHAPTER

Stack Name:	**Tangram**
Folder:	Fun and Games
Size:	130K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Beginner
Special Requirements:	None

11

Tangram: An Age-Old Game Goes Digital

The game of Tangrams, according to Samuel Loyd's 1903 book, *The Eighth Book of Tan,* had been played in China "for upward of 4,000 years, somewhat in the nature of a national pastime." Loyd continued to describe the game: "It consists of seven flat pieces of wood, cut upon the geometrical angles of 45 and 90 degrees, with straight edges which fit together so as to produce a variety of changes which baffles the science of mathematics to compute."[1]

The object of the game of Tangrams is to reproduce a picture, using all seven pieces, or *Tans*. All pieces should be touching, but not overlapping each other. After you get good at reproducing shapes, the object of the game is to tell a story, using the pictures created by the different arrangements of the Tans. Loyd's books include no less then 600 different figures.

The Tangram stack lets you play this game of antiquity and has the modern benefit of instantly arranging the pieces in the proper solution if you get stuck.

How to Use This Stack

To play the Tangram stack:

1. Open the stack. After a short graphic and an oriental tune, you see the main game card, shown in figure 11.1.

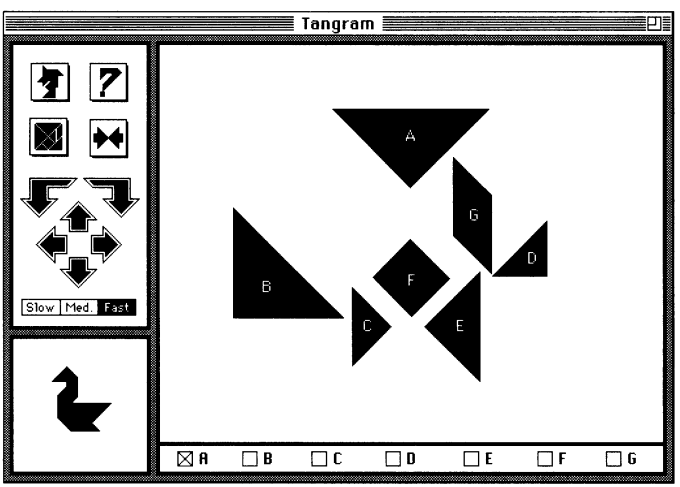


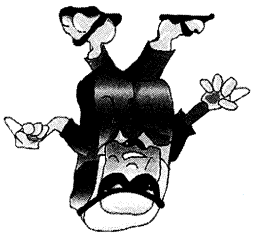
Figure 11.1
The Tangram stack main game card.

   Click the button with the puzzle-shaped icon at the upper left. This button takes you to a series of cards with shapes to choose from.

2. Choose a shape to re-create with your pieces by clicking the shape you want. Figure 11.2 shows some shapes you can choose from.

   You now return to the game card, with the shape you chose pictured at the lower left.

3. Now you must arrange the pieces to duplicate the shape you chose. At the bottom of the card are check boxes. The check box with an "X" in it selects the



piece that you affect with the controls at the left. To choose a shape to move or rotate, click one of the check boxes.

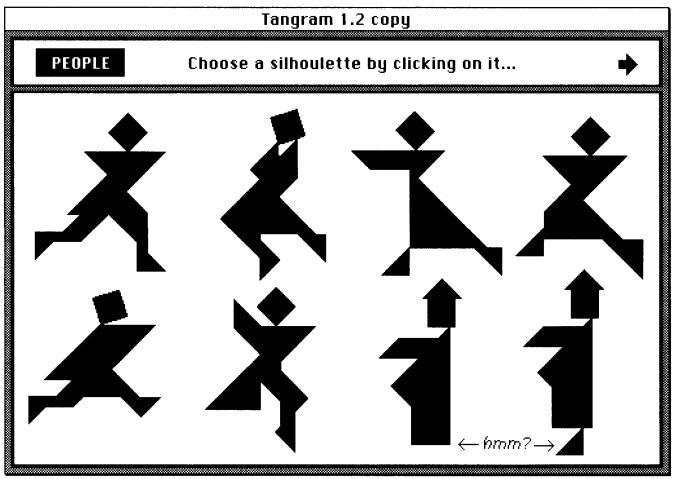


Figure 11.2
Shapes you can try to reproduce in the Tangram stack.

4. Click and hold down the mouse on one of the arrows at the left to move your chosen piece left, right, up, or down. The buttons **Slow**, **Med.**, and **Fast** control the speed at which the piece moves in response to the controls. It's probably best to start with a fast move and then use the slow setting to tweak your position, but you can judge which strategy is best for you.
5. Click the bent arrows to rotate your chosen piece forty-five degrees. Keep clicking it until the piece is rotated the way you want.
6. Repeat Steps 4 and 5 until your large picture looks like the small picture.
7. Think you've got the correct arrangement of pieces for the shape you chose? Check your answer by clicking on the icon with the two arrows facing each other, which puts the chosen shape together.

This stack once again shows the power of HyperCard to simulate real-world objects. Besides the instant-answer feature, HyperCard can add even more to this game. With the addition of new networking features in HyperCard 2.1 and System 7.0, this stack could easily become a collaborative or competitive game on a network. Moves by one player would be echoed to another copy of the stack running on another Macintosh. A stack that lets you play a game of Battleship between two Macintoshes has already been created with HyperCard to show the power of HyperCard and AppleEvents. Size considerations kept the HyperBattleship game out of this set of stacks. If you are interested in an example of how AppleEvents work in HyperCard, take a look at the HyperConference stack, also in this collection.

How Is It Done?

How does the Tangram stack move those pieces around? This stack uses a custom Tangram font. When you click to see the correct answer, a script moves all the text fields that hold the Tangram characters to the correct positions. Then the script places the proper Tangram font characters, each representing the appropriate rotation of the game pieces, into all of the fields. Pretty clever, eh? This idea could be used in other game stacks. What if the font characters represented chess pieces? Or, how about a horse race stack game using a special horse font? The horses could each gallop across the screen in long text fields, and your race could even include the starting bugle and the roar of the crowd!

You can use fonts that have characters other than letters to produce impressive animation in HyperCard. If you are interested in creating custom fonts, there is a marvelous product for making screen fonts called Fontastic from Altsys. As is often the case with HyperCard, the main limit is your imagination.

[1] *The Eighth Book of Tan*, originally published in 1903 by Loyd and Company, and reprinted by Dover Publications in 1968.

CHAPTER

Stack Name:	**The Pizza Navigator**
Folder:	Fun and Games
Size:	60K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Beginner
Special Requirements:	None

12

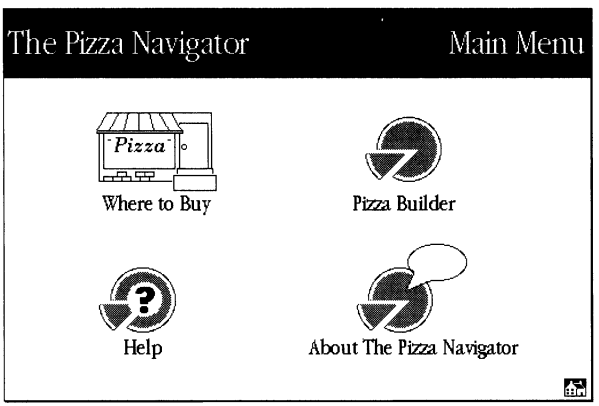
The Pizza Navigator: A Stack to Go

This stack is just for fun and digestion. It definitely would be entertaining at a party, and certainly will bring a smile to all but the most humorless computerphobes. HyperCard lets you try out some software "what ifs." For example, what if you could program your VCR from your computer? Or, what if you could use your Macintosh to tune up your car? Better yet, what if you could order a pizza from your computer? This stack is a whimsical answer to that last "what if."

How to Use This Stack

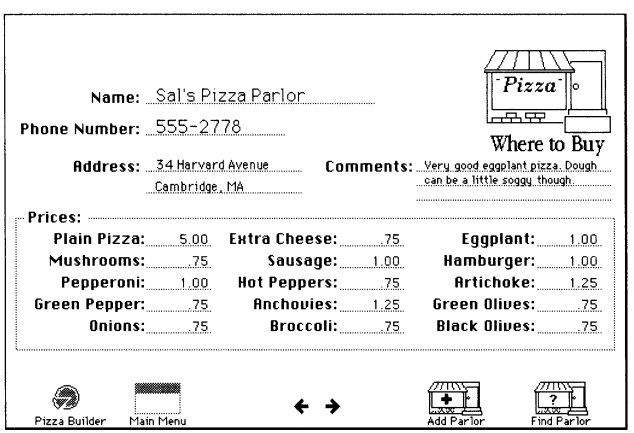
The Pizza Navigator contains three kinds of cards. There is a main menu card, shown in figure 12.1, with buttons labeled **Where to Buy**, **Pizza Builder**, **Help**, and **About The Pizza Navigator** to take you to these screens.

Figure 12.1
The Pizza Navigator main menu.



Where to Buy is where you list your favorite pizza parlors, the prices they charge for a plain pizza, and the cost for added ingredients. There is also space for the address, phone number, and any comments about the pizza or service. Figure 12.2 shows a typical Where to Buy card.

Figure 12.2
A typical Where to Buy card.



The fun really starts at the Pizza Builder, shown in figure 12.3.

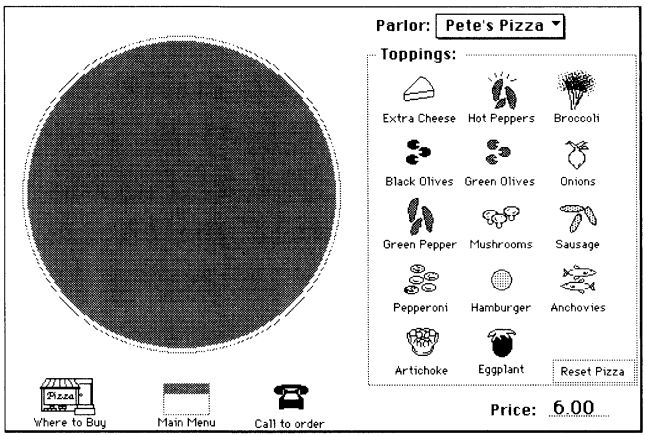


Figure 12.3
The Pizza Builder screen.

The Pizza Builder is where you assemble your pizza and find out how much it costs. To build your pizza, drag each topping that you want on it from the right to the pizza at the left. After you drag each topping, the price of the pizza, shown at the bottom, goes up. The price is based on the charges for base price and toppings for the parlor listed at the top.

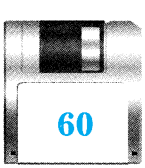
If you want to see what this kind of pizza costs at another parlor, click and drag the pop-up menu to choose another parlor from the list. The Pizza Navigator recalculates your price depending on the parlor you select.

To remove all of the toppings at once and start from fresh, click the **Reset Pizza** button at the lower right. After you figure out the way you want your pizza, you can dial the parlor using the Mac speaker by clicking the phone icon at the bottom. You already know the price.

Where This Stack Doesn't Cut It

The Pizza Navigator shows that the Macintosh user interface can be applied to all sorts of tasks. It also shows where this particular interface might not be up to the job. Consider the Pizza Navigator's limitations: It can't address price differences between large and small pizzas, special prices for specific ingredient combinations (such as a vegetarian or all-meat pizza), or that certain ingredients might only be available at one parlor. It might be nice to fax your order or save frequently chosen pizza topping combinations for easy recall (if you want to try to add these features, be my guest). The most significant problem, though, is that the real world usually doesn't have such an orderly and regular pricing scheme for pizzas; each parlor differs enough to stretch the limits of a computer's flexibility.

As our lives become more complicated, you will probably see bankers, store owners, restauranteurs, and other merchants make their goods and services more accessible to computers, but we have a long way to go. Currently you can shop and pay bills from your Macintosh by accessing on-line services with a modem. You also can control your house's lights and heat, and even use your Mac as a digital answering machine. To this day, however, you can't easily order a pizza with your Mac. Maybe it's just as well—I'd probably gain 20 pounds if I could.



CHAPTER

Stack Name:	**Bond of Union**
Folder:	Fun and Games
Size:	82K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.0
Level of Expertise:	Beginner
Special Requirements:	None

13

Bond of Union: A Digital Interpretation

The original artwork *Bond of Union* was created by Dutch artist M.C. Escher. It shows the heads of a man and woman, articulated by a winding Mobius strip and accompanied by floating spheres. Like many images by Escher, *Bond of Union* is probably familiar to you. His drawings have been widely reproduced and are particularly fascinating to those who are interested in mathematics and computers.

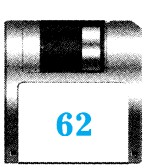
This version of *Bond of Union* takes a 3-D image and makes it into a 4-D image. How? If you consider the fourth dimension to be that of time, this stack takes a scan of Escher's image and animates it, producing a strange and compelling piece of computer art.

How to Use This Stack

To view the animated version of this work of art, simply open the stack. The animation continues until you click the mouse. If it takes a moment to stop, be patient. To repeat the show, click the mouse when the message box instructs you that you may.

Computer Art

This stack demonstrates that the card-based structure of HyperCard stacks makes them a natural for flip-card style animation. Because HyperCard can move from card to card quickly, you can easily create this illusion of motion. In addition, this stack suggests that HyperCard is a great platform for computer-based art. With digitally stored and delivered sound, text, and graphics, new mixed-media creations including poetry, drawing, and music are possible. In addition, because HyperCard provides nearly instant access to any part of a stack at any time, stacks can be nonsequential works of art in which the viewer roams freely through the piece. This kind of stack truly has no beginning or end. In fact, if you like, start up the stack and switch to another application in MultiFinder (or System 7.0). The image continues to cycle, providing an animated backdrop to your work. Too bad it takes up so much RAM.



CHAPTER

Stack Name:	**HyperConference**
Folder:	Utilities
Size:	48K
Type of Mac:	Any Mac that runs System 7.0
RAM Installed:	2M
System Version:	7.0
HyperCard Version:	2.1
Level of Expertise:	Intermediate
Special Requirements:	The System 7.0 features of File Sharing and Program Linking available on each machine, an AppleTalk network of at least two Macintoshes, and two copies of HyperCard 2.1.

14

Fast Talk, or HyperConference

HyperConference is one of a few stacks in this collection that require HyperCard 2.1 and System 7.0. If you have these HyperCard and System versions and have more than one Mac connected to a printer, you have a network that can support real-time, computer-based conferencing. This means that as you type text on your screen, you can send it almost instantaneously to another person on a computer on your network, similar to a teletype. HyperConference uses some of the built-in capabilities of HyperCard 2.1 and System 7.0 to communicate over your AppleTalk network.

Getting Ready to HyperConference

HyperConference uses the System 7.0 feature called *Program Linking*. Program Linking enables applications to communicate with each other, either on the same Macintosh or over a network. To use HyperConference, copy the stack to any Macintosh you wish to converse with, and then check the Sharing Setup control panel found in your system folder, as shown in figure 14.1.

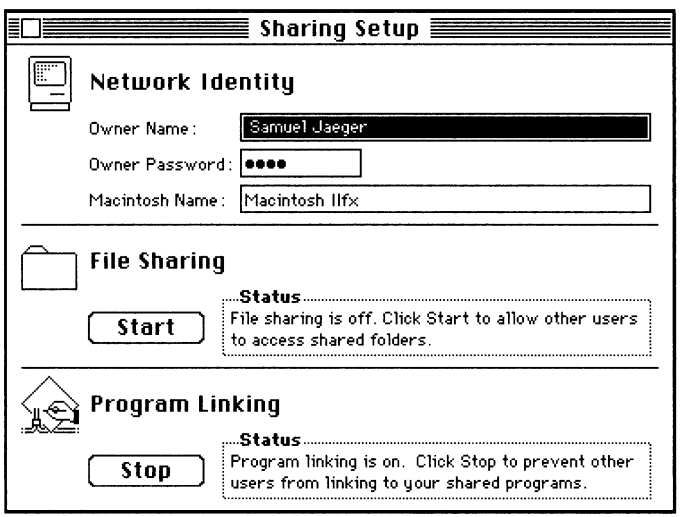


Figure 14.1
The System 7.0 Sharing Setup control panel.

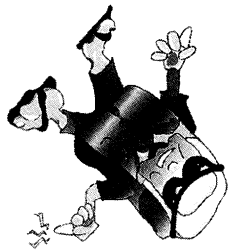
Even though it really has nothing directly to do with sharing files, the button to control Program Linking is at the bottom of this control panel. To use HyperConference, you must have Program Linking turned on.

In addition, if you have never done any file sharing or program linking before, you should have your Users and Groups set up so that people on your network can be identified, and so that each has a password to link to files and applications on your Mac.

To set up new users, open the Users and Groups control panel, shown in figure 14.2.

From the **File** menu, select **New User...**. Name the new user the same way you would name a new folder (click the name below the icon and start typing) and then double-click the icon. You should see something like figure 14.3.

Although it is not absolutely necessary, you really ought to give your new user a password. If you don't, anybody using his or her name could link to the same files. The control for Program Linking is once again at the bottom of the window. Click the check box to allow this user to link to applications on your Mac.



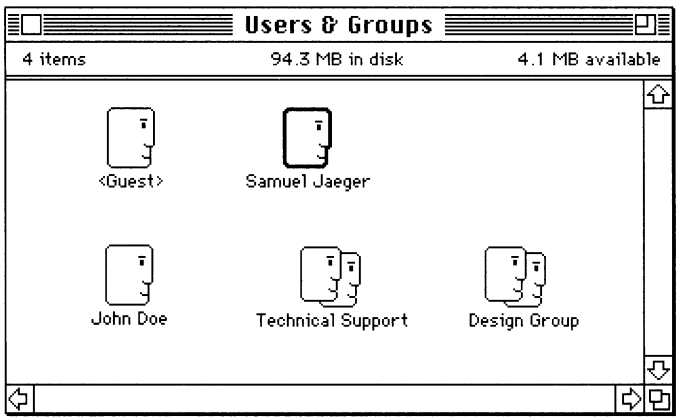


Figure 14.2
The System 7.0 Users and Groups control panel.

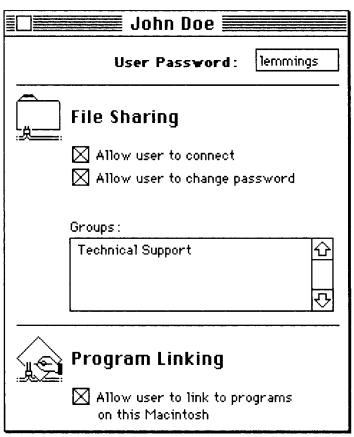


Figure 14.3
The User Information window.

Your fellow conferencer should do all the above as well; otherwise, you are likely to have a very one-sided conversation.

How to Use This Stack

If you've made it this far, you are (finally!) ready to chat over the network. If not, recheck the control panels mentioned earlier. Fortunately, HyperConference is much easier to use than it is to set up. Start up HyperConference, and have your cohort do the same with his or her copy.

The HyperConference chat happens on one screen, shown in figure 14.4.

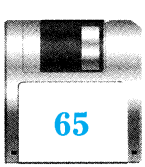
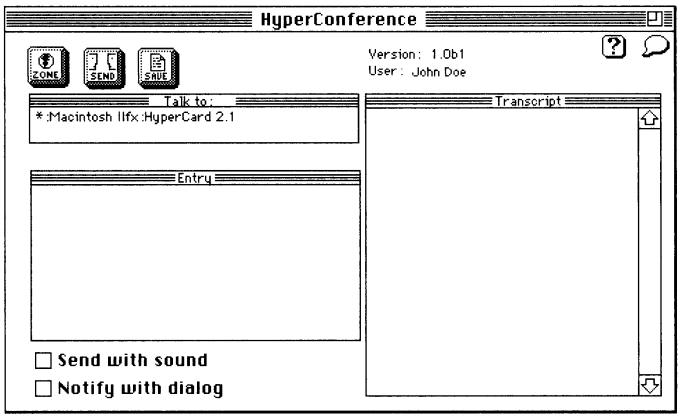
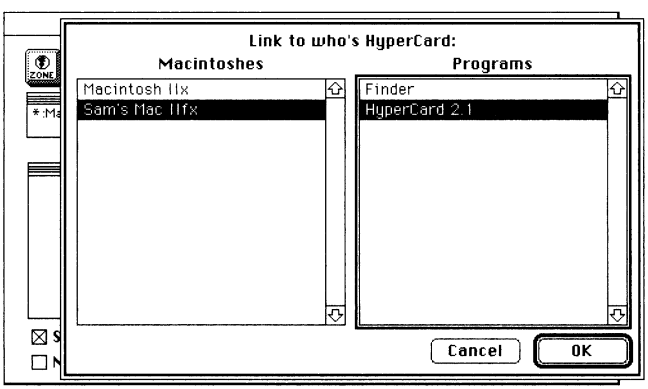


Figure 14.4
The HyperConference main screen.



To choose with whom you will chat and who will show up in the Talk to: field, click the button on the left labeled **ZONE**. You see a dialog box that looks like figure 14.5.

Figure 14.5
The Program Linking setup dialog.



Choose the Macintosh to which you would like to connect from the list on the left. The list on the right then shows the applications running on the Mac to which you can connect. One of them is the Finder, and the other (hopefully) is HyperCard 2.1. Click **HyperCard 2.1** (or simply **HyperCard**, if that is what they named it). Click **OK**.

If you made it this far, you're almost golden. If not, try clicking the **ZONE** button again and be sure to choose HyperCard on your partner's machine (not your machine, or you end up just mumbling to yourself). Type something into the Entry field. `Hello` is fine. Then click the **SEND** button at the top or press Enter (the Return key won't send your message, because you can send more than one line at a time.) The first time you do this, you will see the Program Linking dialog box, shown in figure 14.6.

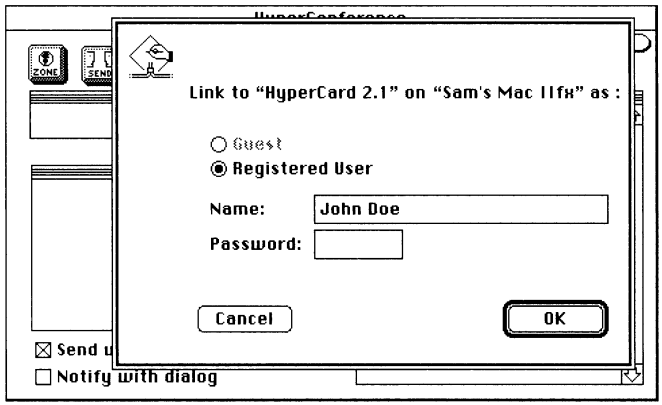


Figure 14.6
The Program Linking Password dialog box.

Did your partner tell you the password to link to his/her machine? Type in your password and click **OK**.

Your text appears in the Transcript field. After a second or two, your friend should see the text you sent. He or she can type back the same way. When you receive messages, the messages are also recorded in the Transcript field. You did tell your partner his or her password, didn't you? If so, he or she can fill in the password dialog box (once) and respond to you. Now you are ready to chat without any further interruptions.

A Few Extra Notes

There are some check boxes at the bottom of the screen with the options **Send with sound** and **Notify with dialog**. If you would like a small ringing sound to accompany your messages, make sure that this check box is checked. HyperConference can stay open in the background while you work on other tasks. If the **Notify with dialog** button is checked, HyperConference alerts you when you receive a message, with a dialog box like the one shown in figure 14.7.

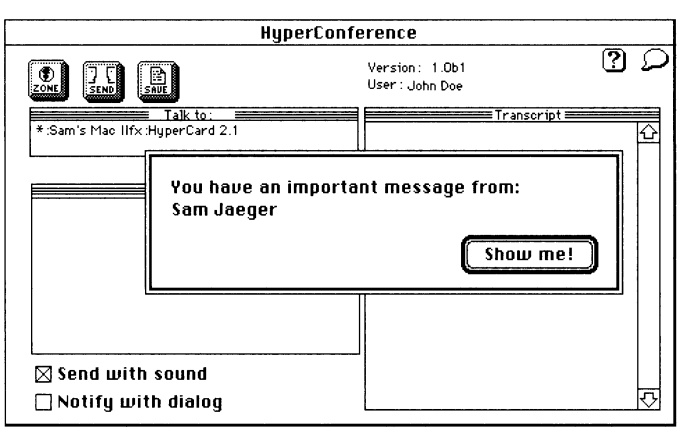


Figure 14.7
HyperConference Notification when running in the background.

For the **Notify with dialog** feature to work, your HyperCard application must be called *HyperCard*, not *HyperCard 2.1* or *Fred's HyperCard*. This is because HyperConference is looking for just the word *HyperCard* on the application menu at the upper right to bring HyperCard to the front.

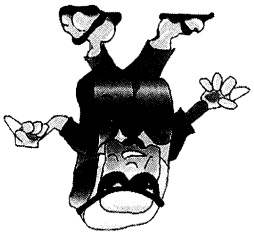
You can save the transcript out to a text file, which might come in handy if you need to document a conference.

More than Just Chats

HyperConference is actually a simple demonstration of the brand-new features of System 7.0 called AppleEvents. AppleEvents are messages that can be sent from one application to another, either on the same machine or to other machines on your network. System 7.0 provides the "wiring" for AppleEvents to travel on, and HyperCard is one of the first applications to support the wiring. HyperCard talking to HyperCard could produce shared address books and calendars, as well as a powerful paging system. It would not take many changes to HyperConference to create a screen with 10 to 15 buttons on the screen, each one able almost instantly to page with a text or sound message another Macintosh in another office. To modify HyperConference, take a look at the scripts of the **SEND** button and the stack script. Most of the harder stuff already has been figured out for you.

Final Notes

The interface of HyperConference is similar to CE Software's QuickMail, which supports chats between more than two users on your network. Also, to send a quick AppleTalk message in the form of a dialog box, there is a shareware chooser extension called *Broadcast* that works in both Systems 6 and 7. Some offices use it to send the latest joke over the network.



CHAPTER

Stack Name:	**Balloons**
Folder:	Utilities
Size:	84K
Type of Mac:	Any Mac that can run System 7.0
RAM Installed:	2M
System Version:	7.0
HyperCard Version:	2.1
Level of Expertise:	Advanced
Special Requirments:	To see color pictures in help balloons, you need a Mac that can display color and a color monitor. UserLevel must be set to 5.

15

Balloons: Inflatable HyperCard Help

One of the most visible changes to System 7.0 for the Macintosh was a new kind of help facility called *Balloon Help*. Balloon Help labels what you see on the Macintosh screen without interfering with your work. The balloons from Balloon Help appear only when **Show Balloons** has been selected from the Balloon Help menu at the top right of the System 7.0 menu bar. Figure 15.1 shows the icon for the Balloon Help menu.

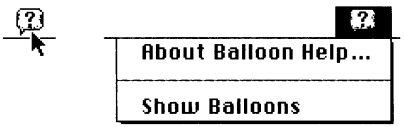


Figure 15.1
The Balloon Help menu icon and the Balloon Help menu.

If you've used System 7.0 for more than five minutes, chances are you turned on Balloon Help for a while, just to see how it works.

Balloon Help is most useful when you first explore a new application. If you are like many Mac users, you don't read the manual. If this application is worth working with, some of the tasks must be clear without having to read how to use them, right? As intuitive as the Macintosh interface is supposed to be, chances are there might be something that isn't entirely self-evident when you first lay eyes on it. The object could be an icon on a tool palette. It could be some strange-looking control in a dialog box. That's the time when Balloon Help comes to the rescue. Balloon Help can explain what a particular item is just by showing some words of explanation while your cursor is pointing to the object in question. Figure 15.2 shows what happens when you turn Balloon Help on and point to the title bar of a window.

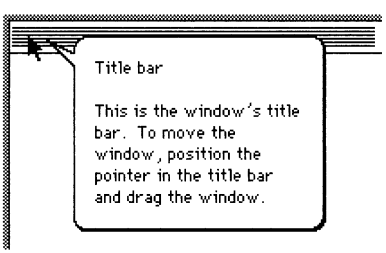


Figure 15.2
Balloon Help for the title bar of a window.

Balloon Help is not the result of the Mac figuring out everything for you and displaying written explanations. Balloon Help is a collection of definitions created by the software's designer. For balloons to appear in an application, the developer must program into the system the message that help is available for each item, write the help text, and insert the text into each balloon.

Because we come into contact with so many HyperCard stacks that often do unusual and surprising things, Balloon Help really comes in handy for HyperCard stacks. Fortunately, this stack not only makes it possible to add Balloon Help to your stacks, it even makes it easy.

How to Use This Stack

When you open the Balloons stack, you see a main card with buttons to all of the subjects covered in the stack. Notice that all of the screens were designed with an attractive graphic flair. Figure 15.3 shows the main card of the Balloons stack.

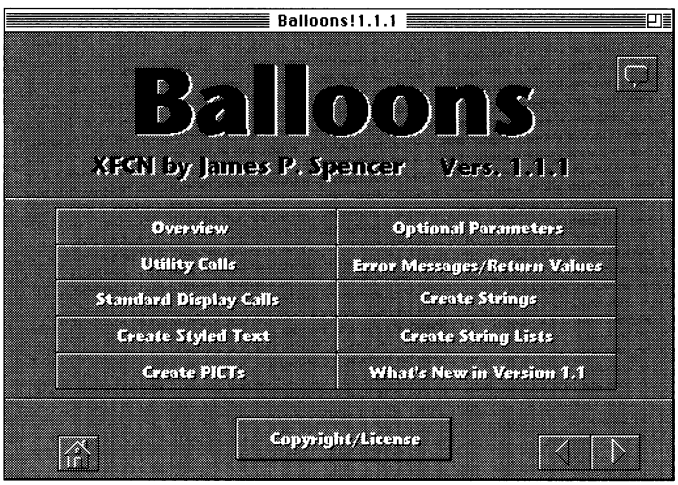


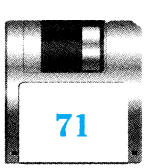
Figure 15.3
The main card of the Balloons stack.

The code to alert the system to show a balloon is built into a HyperCard XFCN called Balloons. An XFCN is an external function and is almost the same as an external command. The difference between an XFCN and an XCMD is that an XFCN returns a value back to HyperCard. In the case of Balloons, the returned value indicates whether the help balloon was successfully shown. Using an XFCN also helps HyperCard determine whether the stack is running under System 7.0, the only system which has Balloon Help.

You can move the Balloons XFCN and the capability of showing Balloon Help into any of your stacks by using a resource editor like ResEdit. When the Balloon XFCN is in the resource fork of your stack, it adds the word *Balloons* to the vocabulary of HyperTalk that your stack understands. To show a balloon from a button with text saying "Hi there, this is a nice button," you use the following command:

```
get Balloons ("Show","Hi there, this is a nice button", the loc of me)
```

The real trick is when to show the balloon. This requires a bit of explanation on system messages in HyperCard.



Mouse Messages

You have already seen scripts that begin with the words `on mouseUp`. HyperTalk scripts respond to messages. A response to a message in HyperTalk is called a *handler,* because it handles a message. In the case of `on mouseUp`, the handler responds when, first, the mouse button goes up from a down position while the mouse cursor is over the button with the mouseUp script and, second, the handler receives the system message `mouseUp`. In case you were wondering why the message in most button handlers is `mouseUp` and not `mouseDown`, that's because the Mac lets you change your mind before committing to a mouse click on most buttons. Try it on the close box of a window. See? The window doesn't close until the mouse button goes up after you have clicked down.

There are other messages that HyperCard can receive from the mouse. Any time the cursor is moved over a button, HyperCard gets the message `mouseEnter`, and any time the cursor moves away, HyperCard gets the message `mouseLeave`. Usually these messages are ignored because there is no handler that is set to execute when these messages are sent. But `mouseEnter` and `mouseLeave` indicate the perfect times to use the Balloons XFCN; those are the times when a balloon shows up and disappears with other applications.

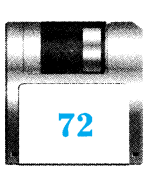
So in addition to the `mouseUp` handler in a button using Balloon Help, you would also see a `mouseEnter` and a `mouseLeave` handler. A typical example of a button with Balloon Help would look like this:

```
on mouseEnter
   get Balloons ("Show","This button goes to the main menu", the loc of me)
end mouseEnter

on mouseLeave
   get Balloons("Remove")
end mouseLeave

on mouseUp
   go to card "Main menu"
end mouseUp
```

The `mouseEnter` handler shows the balloon, the `mouseLeave` handler removes the balloon, and the `mouseUp` handler does the work that the balloon help is talking about. It doesn't matter what order these handlers are in, they all just need to be in the button's script window. In case you were wondering about the phrase `the loc of me`, this bit of HyperTalk says to show the tip of the balloon from the location of whatever object is currently "me," which in this case is the main menu button.



Bigger Balloons

The Balloons stack also comes with demonstrations and information about more ambitious uses of the Balloons XFCN, including balloons that show color pictures and styled text. The Finder can't even do that! If you are interested in delving into the details of more advanced uses of this XFCN, check out the stack, which has a great deal of built-in documentation (and balloons). A larger, full version of the stack includes many authoring tools and tips for creating your own pictures and text for display in help balloons. If you are interested in getting this version, there is information on how to get in touch with the folks who created Balloons, HyperActive. There wasn't enough space to include it here, but the stack is free if you send them a disk.

CHAPTER

Stack Name:	Say the Numbers
Folder:	Utilities
Size:	64K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Intermediate
Special Requirements:	UserLevel must be set to 5.

16

Say the Numbers: The Number You Have Dialed...

Say the Numbers is small, both in screen area and size. The stack has one purpose in life: It does what its name says; it speaks numbers using a digitized voice.

How to Use This Stack

To see how the stack works, open it. Say the Numbers should look like figure 16.1.

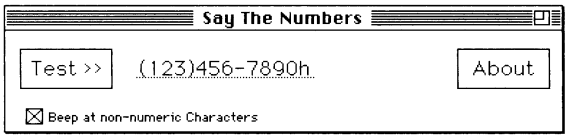


Figure 16.1
The Say the Numbers stack.

Type a number into the field and click on the **Test >>** button. You should hear a voice saying the numbers you typed in. If you type any character that is not a number, Say the Numbers skips over it with a short pause.

So (Say) What?

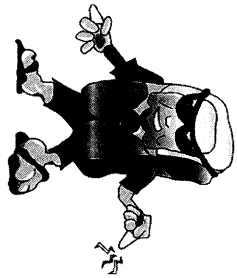
At first glance, Say the Numbers is a bit cute, but is it useful for my other stacks? Actually it is, because of a feature that was added to HyperCard 2.0: the command

```
start using <a stack's name>.
```

As you already know, HyperCard stacks have a structure of cards, within a background, within a stack. Each of these parts of a stack can have a script attached to it. The same way that paint graphics, fields, or buttons on a background screen layer apply to all of the cards for that background, any script typed into the background applies to all the cards for that background. Furthermore, a stack can contain a script, and the script can apply to every card, in every background, within that stack. Therefore, a script that translates a series of numbers to a set of sounds and plays them can be placed in the stack script. There it is available as long as you are in that stack. This relationship of scripts and their placement is called the HyperCard hierarchy. If you make good use of the hierarchy, you can achieve more with less scripting, and make stacks more customizable and easier to change.

Multi-Stack Scripts

What if you want this number-to-sound capability in more than one stack? Before HyperCard 2.0, you either had to paste the scripts and associated sound resources into every stack that used them, or paste the scripts and sounds into your Home stack. Home stacks got very cluttered with scripts, XCMDs, and sounds, and people frequently tried to install them into other people's Home stacks so that their multi-stack systems would work. Messing around with other people's Home stacks is not a polite thing to do, to say the least! Then came the wonderful *start using* command. This command attaches a stack to the HyperCard hierarchy, and lets all of the scripts and resources of that stack be used as if it were the stack you currently have open.



Making Say the Numbers Work for Your Stacks

If you would like to use the number-speaking capability of Say the Numbers in your address book, or in any other stack that involves entering numbers, you need to add some scripts to that stack to use Say the Numbers. Here's how:

1. Open the stack to which you want to add number-speaking features. From the **Objects** menu, choose **Stack Info...**.
2. When you see the **Stack Info...** dialog box, click the **Script...** button.
3. You are now editing your stack's stack script. You should see a text editing window with the title "Script of stack (your stack's name)". Type the following into the script editing window:

```
on openStack
   start using stack "Say the Numbers"
end openStack
```

   This is a handler (handlers were covered in chapter 15) that handles the message your stack gets when it is opened. Also, be sure there is not already an `openStack` handler in your stack script. If there is, make sure `start using stack "Say the Numbers"` is the second-to-last line of the handler (before `end openStack`).

4. Also, just to be nice and tidy, you should include the following:

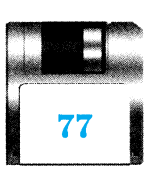
```
on closeStack
   stop using stack "Say the Numbers"
end closeStack
```

   This handler releases Say the Numbers from its duties when you close your stack and frees up some memory for you to use as well.

   OK, now close your stack, and then open it to send the `openStack` message. Then, bring up the message box by choosing **Message** from the **Go** menu and type

```
saytheNumbers "1234"
```

   You should hear the voice saying "One, two, three, four," just as it did when you clicked on the **Test >>** button. If not, check the scripts of your stack again, as well as what you typed into the message box. If you heard the voice, congratulations! You have successfully added a stack to the HyperCard hierarchy.



Where Would This Come In Handy?

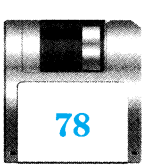
Say the Numbers has two definite uses. One use is to teach children numbers; another use is for adults. After you enter some numbers into a field, perhaps from dictation from the Information operator, Say the Numbers can read back to you what you typed so that you can check it. To do this, place a button in your stack with the following script:

```
on mouseUp
   saytheNumbers background field "Phone Number"
end mouseUp
```

Here, background field "Phone Number" would be the field you want Say the Numbers to say. If you want Say the Numbers to beep at non-numeric characters (not including the dash or space), use this script:

```
on mouseUp
   saytheNumbers background field "Phone Number",beep
end mouseUp
```

Say the Numbers demonstrates the power of modular software. In the future, you'll see more and more Macintosh software that is small and performs just one or two tasks. These forthcoming small applications will talk to each other, similar to the way that HyperCard sends messages (like `saytheNumbers`) from one stack to another to provide the programming and resources to handle the message. This little stack might be a taste (or sound) of things to come.



CHAPTER

Stack Name:	**PICS Player**
Folder:	Building Blocks
Size:	72K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Advanced
Special Requirements:	To see color animations, you need a Macintosh that displays color and a color monitor. UserLevel must be set to 5.

17

PICS Player: A New Way to Get Animation into HyperCard

The PICS Player stack contains an external command called *playPICS*. This external command adds the word *playPICS* to the HyperTalk language, giving HyperCard the ability to play a color or black-and-white PICS animation. A PICS animation is a Macintosh standard file format that many applications can read and write to disk. PICS files are a standard interchange format for displaying frames of an animation and are often exported by expensive and sophisticated 3-D rendering and animation applications.

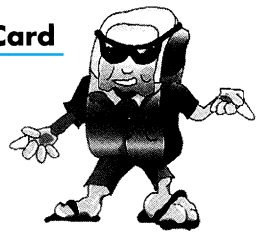
If you have worked with the Mac for even a short time, you've probably encountered Macintosh standard formats already. One well-known standard format for text is ASCII, which is plain text with no formating. All word processors import and export ASCII. Another format is RTF (Rich Text Format), which includes some style and font information for the text and is supported mainly by Microsoft Word. MacWrite, an early Mac standard for word processing, includes margin and tab settings, as well as embedded graphics, and is read and written by many word processors. The most popular graphic formats for the Mac include MacPaint, Tagged Image File Format (TIFF), and the most widely distributed graphic format of all, PICT. The PICT format includes object and paint information, and supports color data at a wide range of bit depths, from black-and-white all the way to 24-bit color.

A PICS file is merely a series of PICT frames (PICT resources, to be exact) that applications may easily export or import to display or edit as an animation. Some of the applications that support the PICS file format include MacroMind Director, Swivel 3-D by Paracomp, Silicon Beach's Super 3D, and AddMotion by Motion Works. There are many more applications not listed here, and most of them export PICS files. If you have never seen a PICS file, two small ones are included with the PICS Player stack as examples. Once you have a PICS file and the PICS Player stack, it's showtime!

How to Use This Stack

When you open the PICS Player stack, you immediately see an example of the playPICS XCMD in action. Figure 17.1 shows a frame from the first demonstration. Obviously, a book can't include the actual animation, so you'll have to open the stack to see the real thing in motion.

After you click the mouse, you see the Theatre screen. There are some examples of the playPICS XCMD, and a button that takes you to more information called **About PICS Player**. Figure 17.2 shows the PICS Player Theatre.



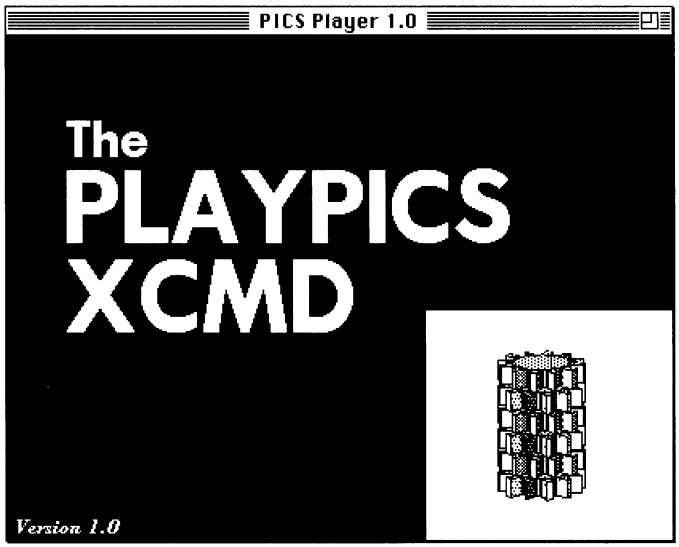


Figure 17.1
The first card of the playPICS external command stack.

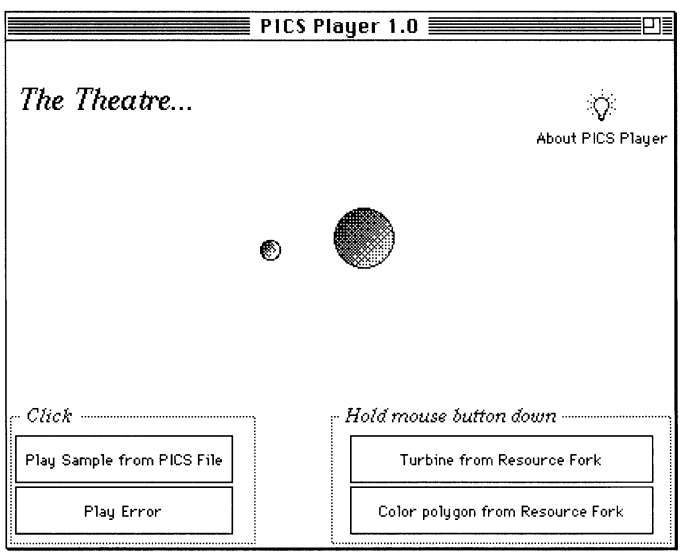


Figure 17.2
The PICS Player Theater card.

Notice that some of the buttons include the words *Resource Fork*. (For information about the resource fork and what it can do for you, see Chapter 5, "About External Commands.") The resource fork of this stack contains some PICT resources. The playPICS XCMD can also play a series of sequentially numbered PICT pictures in a stack's resource fork instead of in a PICS file. The sequential numbers of PICT pictures in the resource fork are IDs which may be changed with ResEdit. There are two advantages to using PICT resources

to play an animation. First, if you give your stack that includes an animation to someone else, you don't have to worry about whether they have the required external animation files in the proper folders accompanying the proper stack. Second, PICT resources may be preloaded into RAM without the user knowing. Therefore, you can trigger an animation immediately when you click a button, giving the user speedy feedback. When you preload an animation into RAM, the animation has been effectively cued up and is ready to appear with very little delay.

If you would like to see an animation play from a sample PICS file, there are two included with the stack. Click the button **Play Sample from PICS File**. You see a standard open file dialog box where you can select a sample PICS file. The file you choose is used for the rest of the session.

If you would like to switch examples during a session, hold down the option key as you click the Play Sample from PICS File button to see the open file dialog box again.

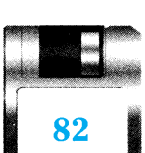
Adding PICS Animations to Your Stacks

To add PICS animations to your stacks, first you must copy and paste the playPICS XCMD into the resource fork of your stack. Once you do this, your stack can play any PICS file on the card window. The way that you play a PICS animation from a script is as follows:

```
on mouseUp
   playPICS fileName
end mouseUp
```

This script plays the animation with everything set to standard defaults. The defaults include a starting ID (not applicable because this is a file, not a set of PIC resources); an ending ID (same situation); a location on the card of 0 pixels from the left and 0 pixels from the top; a speed of 60 frames per second; and instructions to play the complete animation one time, preload all of the frames before playing, and make the animation look smoother.

The next script can be used as a model for animations where you might want to change the defaults. Type this script in, or copy it from the button script of the button **Play Sample from PICS File**:



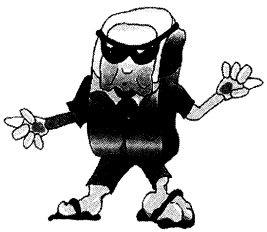
```
on mouseUp
  put myAnimation into whichAnimation
  put "" into startID          -- default 128
  put "" into endID            -- default 127 + number of PICT
  put "" into xLoc             -- default 0
  put "" into yLoc             -- default 0
  put "" into framesPerSe      -- default 60
  put "3" into times
  put "" into preLoad          -- default TRUE
  playPICS whichAnimation,startID,endID,¬
  xLoc,yLoc,framesPerSec,times,preLoad
end mouseUp
```

Then, substitute the full path name of your animation for `myAnimation` and the number of times you would like to repeat the animation for the number `3`. You can also use a field to hold the path name for your animation; use the name of the field in the script if you want to change animations easily. This is the method the Play **Sample from PICS File** button uses to switch animations quickly and easily. If you are interested in other ways of using this versatile XCMD, there is more information available in the "Syntax" section of the PICS Player stack.

What to Do with PlayPICS

PlayPICS is best suited to playing small looping animations to demonstrate concepts. Animations of a heart beating or the orbit of an electron are good examples. You can use playPICS to show larger animations, but if you need extensive use of animation in your stacks, you should look into the commercial products AddMotion from Motion Works, and the MacroMind Director HyperCard Driver, both of which offer sophisticated integrated animation in HyperCard. In the case of AddMotion, you can create full-color animations while still in HyperCard.

CHAPTER



Stack Name:	Status Window
Folder:	Building Blocks
Size:	30K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Intermediate
Special Requirements:	UserLevel must be set to 5.

18

A Stack That Answers the Question "How's It Going?"

Whether you like it or not, the Mac sometimes keeps you waiting. The most common times you wait are when you are printing, copying a file, or perhaps downloading a file from an on-line service. In all of these cases, you often see a window with a message about what is going on. The box might say how many pages are left to print, or how long it will be before the file is done being copied or downloaded.

HyperTalk, the programming language of HyperCard, is easy to learn and read. Unfortunately, HyperTalk is also slow. Hence, HyperCard keeps you waiting, too. The best thing to do in the situations where waiting is unavoidable is to give the user a message like those mentioned previously. Status Window makes this possible.

How to Use This Stack

Open the Status Window XCMD stack. Information about the stack appears at the left, and a **Test Drive** button at the right demonstrates the XCMD. Click the **Test Drive** button to see the demonstration. A status window created with the Status Window XCMD looks like figure 18.1.

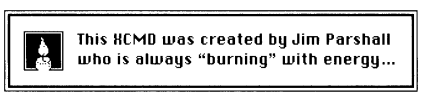


Figure 18.1
A typical status window created with the Status Window XCMD.

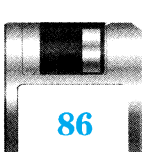
If you are using System 7.0 and are running HyperCard in color, this window has the attractive three-dimensional border that you usually see in System 7.0 dialog boxes.

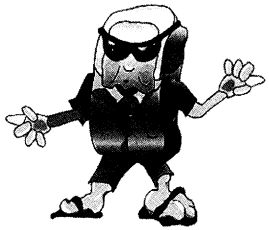
As the demonstration shows, the status window may contain an icon and text. The demonstration with this stack includes an animated icon. The text may also change, so that if you wanted to change the message (for instance, "10 cards to go," "9 cards to go," "8 cards to go," and so on) after the window is showing, you may do so. Although this example includes icons for a flickering candle, you could just as easily use icons that change gradually from one picture to another, like an hourglass filling up.

Working with Locked Screens

Often when a script takes a long time, HyperCard is moving from card to card (or sometimes from stack to stack). If these actions are shown, the script can take two to five times as long. This is because HyperCard has to do the work of updating the screen in addition to the business of the script, which might simply be to retrieve some text. When this is the case, there is a HyperTalk command that can speed up things: `Lock Screen`. This keeps the HyperCard screen from updating while the script is being processed and speeds up the script. Unfortunately, locking the screen also prevents you from seeing the progress of the script or even knowing if it is going to stop!

When you use Status Window to display messages, you may change the text or icon in the status window even if the screen is locked during a long script. As a result, the script runs faster and the user remains informed.





Creating Your Own Status Window

When you move the Status Window XCMD to the resource fork of your stack, the HyperTalk word status window is understood by HyperCard. Therefore, creating a status window is very easy. You can even do it from the message box. Type this into the message box:

```
StatusWindow "©Dome", "test", "Hello there"
```

You should see a status window that looks like figure 18.2.

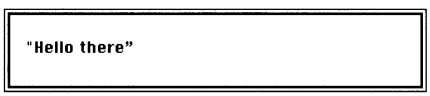


Figure 18.2
Another status window.

The word `"©Dome"` is the creator of this XCMD, and they chose to remind you through the use of their name to bring up the window. `"test"` is the name of the status window. To change or close your status window, you now refer to window `"test"`. Of course, the words `"Hello there"` are the message that appears in the window.

To get rid of the status window, type

```
close window "test"
```

If you would like to include an icon, try

```
StatusWindow "©Dome", "test", "Hello there",1000
```

You should see a status window that looks like figure 18.3.

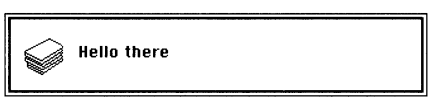


Figure 18.3
A status window with an icon.

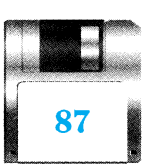
The number `1000` is the ID number of the stack icon that the HyperCard application should be able to display. To change the text of the window, type

```
set "theText" of window "test" to "Hello Mom".
```

To change the icon, type

```
set "theIcon" of window "test" to "1001"
```

This changes the icon to the HyperCard application icon. To find the ID number of the icon you want to use, look in the icon list window, which you can see by clicking the **Icon...** button of the Button Info dialog box.

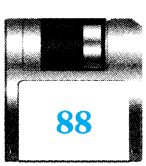


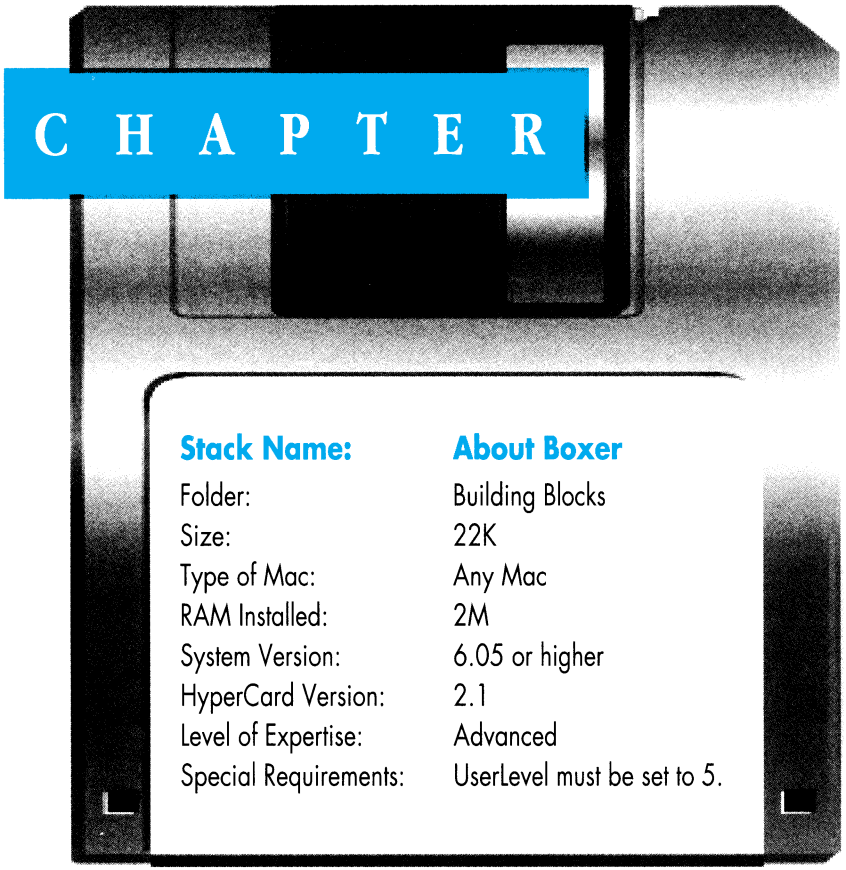
Again, to close the window, type

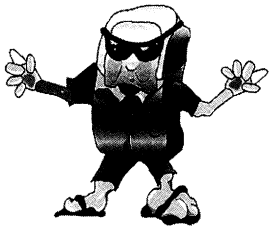
```
close window "test"
```

Other Thoughts

Status Window enables you to keep the user informed about what's going on during a long script. This XCMD also can send instructions to a user without having to use the message box. Furthermore, your script can get rid of the dialog without the user having to click anything. Best of all, Status Window is really simple to use and includes some fun graphic features. If you want to try out your wings by using your first XCMD, Status Window might be the one to try first.





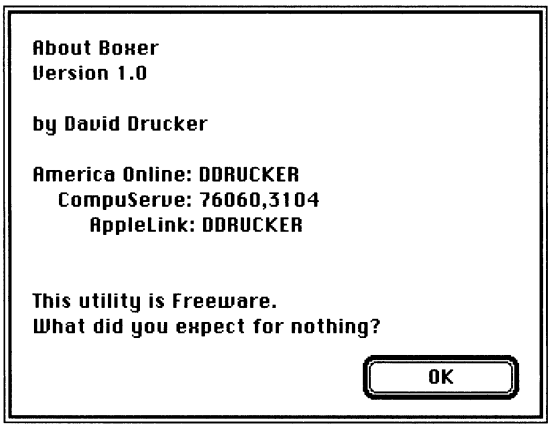


19

About Boxer, for the Egotist in All of Us

About Boxer is a fun widget for creating those dialog boxes that appear when you choose **About** (whatever) from the Apple menu. It also demonstrates a very interesting feature of HyperCard, the ability to create custom-made scripts on the fly. Figure 19.1 shows an example of this kind of about box:

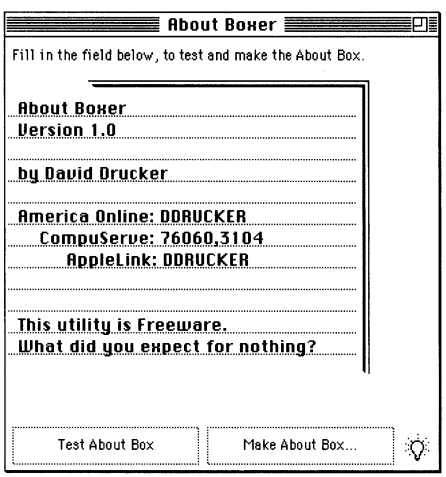
Figure 19.1
A typical about box.



How to Use This Stack

About Boxer is very easy to use. After you see some of the introductory cards, you'll see some on-line documentation. The next card looks like figure 19.2:

Figure 19.2
The about box creation card.



Making an about box with About Boxer is a 3-step process:

1. First, type the text of your about box into the field on the card that looks like a piece of a dialog box (except it has dotted lines in it). If you would like some of your text to be right-justified, you can do this manually by inserting spaces at the beginning of a line to make the ends line up. This is not truly right-justified text,

because the Chicago font is not monospaced (in other words, all characters do not take up the same amount of space). It can look pretty good, though, if you work at it.

2. Click the **Test About Box** button. This shows what your about box will look like after the code to show it is created.

3. Click **Make About Box...**. You then get a dialog box asking if you want to create a **button** or a **handler**.

   Choose **button** if you wish to create a button with all the scripts for displaying your about box. You may copy this button and paste it on either the card or the background of your stack that uses this about box.

   Choose **handler** to create the script to show your about box and then copy it to the clipboard. You would choose **handler** if you wanted to show your about box by choosing it from a menu, or as you first enter a stack (using the `openStack` message that is sent whenever you open a stack).

4. If you do choose **handler**, you will also be prompted for the name of your handler. Be sure to use just one word. If you want to use a handler name that sounds like a few words, the standard convention is to leave out the spaces and capitalize the first letters of new words, like `showMyGreatAboutBox`. This is the word that begins and ends the script. If you choose a word like `showMyAbout`, then the script that is created to show an about box begins with the line

   ```
   on showMyAbout
   ```

   and ends with the line

   ```
   end showMyAbout
   ```

5. You'll see a dialog box telling you that the script has been copied to the clipboard. Go to the stack where you want your about box, paste the contents of the clipboard into the stack script, and save your changes to the stack script.

6. To test your new about box in your stack, type the handler name that you gave the script (like `showMyAbout`) into the message box and press return. You should see your about box appear. There are details for showing the about box from a menu selection, as well as some sample scripts for doing this that you can copy in About Boxer.

7. If you choose **button**, About Boxer creates a button with a special icon that you may then copy and paste to your stack. If you don't like the default icon or want to select your own icon, double-click the button with the button tool selected and then click the **Icon...** button in the Button Info... dialog box.

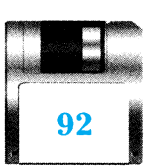
Different Ways to Use This Stack

About Boxer uses the following HyperTalk command:

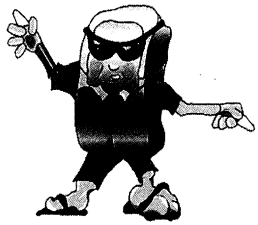
```
set the script of <object> to <a variable containing a script>
```

This feature of the HyperTalk language lets you create a script on the fly, to your specifications, and then attach it to an object such as a button. This is a very powerful concept. Some stacks actually generate complex HyperTalk scripts as you click buttons. This example is a simple one; it takes the text that you type and creates a script to show the text in a dialog box using the HyperTalk command `answer`. There are many more ambitious possibilities, however. Imagine a stack that creates scripts for data entry fields. This stack could accept different kinds of input (only numbers within a certain range, or only a single letter, for example) according to specifications you set as you create the new field. Or, you might create a version of the about box that checks to see if this is the first time a user has ever opened the stack. It would then prompt the user for his or her name, and finally would change its own script to personalize the about box so that the next time that user opens the stack, the about box contains his or her name. Asking for a name and perhaps a serial number is the same way some commercial Macintosh applications personalize themselves out of the box.

Admittedly, the about boxes you create in About Boxer are a bit plain. The Mac has a reputation for having about boxes that really surprise people. The about box for the MacRecorder SoundEdit application has a bomb box that explodes! If you want to create more elaborate dialog boxes in HyperCard, there is a fine HyperCard development tool called Dialoger, available from Heizer Software, that creates color, multi-button, and list dialog boxes for your stacks.



CHAPTER



Stack Name:	Colorizing HyperCard
Folder:	Building Blocks
Size:	176K
Type of Mac:	Any Mac that displays color
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Very Advanced
Special Requirements:	To see color in HyperCard, you need a color monitor and a Mac that displays color. UserLevel must be set to 5.

20

Colorizing HyperCard: A Look at Some Local Color

One of the most common features that appears on the wish lists of people who work with HyperCard is a single word: color. The Colorizing HyperCard stack, although it requires some scripting, brings color patterns and pictures to HyperCard cards and buttons. You can display color on the HyperCard card window itself, as opposed to viewing color pictures in separate windows, a feature that was added to HyperCard 2.0. All of the magic of colorizing HyperCard in this stack is done with a single XCMD called *ColorizeHC*.

How to Use This Stack

Needless to say, in order to see the color in this stack or any other stack that uses the ColorizeHC XCMD, you must have a color monitor and a Macintosh that can display color. The Mac SE/30, Mac LC, Mac IIsi, Mac IIci, and Quadra 700 and 900 all can display color if they have an attached color monitor.

When you open the Colorizing HyperCard stack, you see a black screen with the words *Colorizing HyperCard....* Wait a moment while the XCMD loads the color information into RAM. You then see a color splash screen that includes some art representing crayons. Figure 20.1 shows this card.

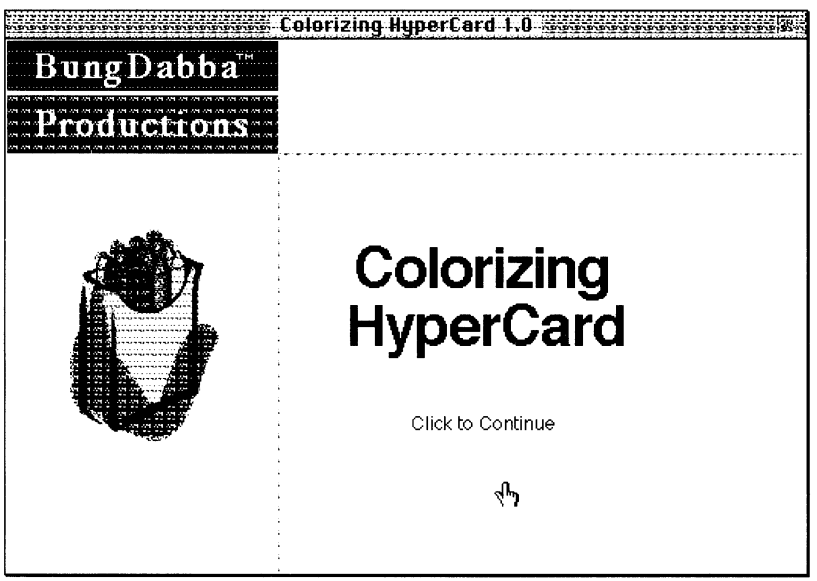


Figure 20.1
The Colorizing HyperCard splash screen.

To continue, click the mouse. You then see a stack disclaimer with some legal and other general information. To the right, there is an index of all the topics available. To go to any topic, click it. One of the best places to click first is the **Stack Overview** at the bottom of the list. Clicking the **About this Stack** button also takes you to the same card.

To quote from the information in the overview:

"The ColorizeHC XCMD is a HyperCard® XCMD that allows HyperTalk programmers to add color 'backdrops' to HyperCard stacks. A 'backdrop' is a color image that is automatically drawn at the same time as HyperCard's card image. It can be used to display images from PICT resources and PICT files, and provides several commands for modifying existing images."

Adding Color Pictures to HyperCard Cards

Probably the most powerful and important use of the ColorizeHC XCMD is adding a color picture to a card. To do this, your script would read something like this:

```
on mouseUp
   put the rect of cd btn "placeToShowPICT" into pictureRect
   colorizeHC "add","French Stamp",pictureRect
end mouseUp
```

`Rect` is short for rectangle. A rect consists of four numbers, or coordinates. The first two are for the horizontal and vertical values of the top and left of a rectangle, and the third and fourth are for the bottom and right coordinates of a rectangle. Rather than use absolute numbers for a rectangle, it is usually more convenient to use the rect of a button or field, so that you can easily move or resize where you want the color picture. Figure 20.2 shows an example of what the colorizeHC script does:

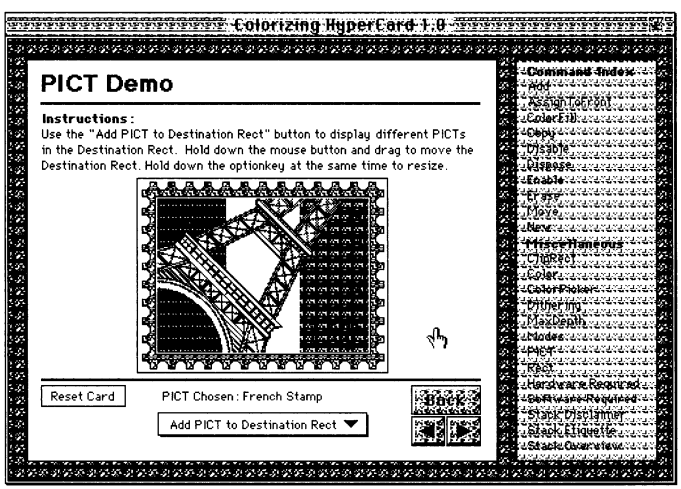
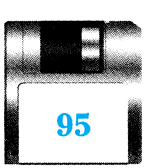

Colorizing HyperCard 1.0
PICT Demo
Instructions:
Use the "Add PICT to Destination Rect" button to display different PICTs in the Destination Rect. Hold down the mouse button and drag to move the Destination Rect. Hold down the optionkey at the same time to resize.
Reset Card
PICT Chosen: French Stamp
Add PICT to Destination Rect
Back

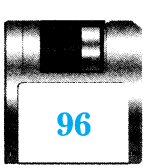
Figure 20.2
An example of the colorizeHC script that adds a color picture to a card.

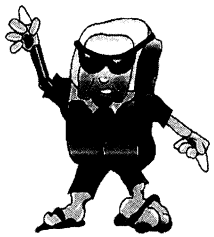
Other Colorful Options

There are many different ways to use the ColorizeHC XCMD, and they are all listed and demonstrated in the Colorizing HyperCard stack. However, if your documents use color a great deal, and you are not absolutely committed to HyperCard, there are two other commercial products that are very much like HyperCard (sometimes they are even referred to as HyperCard clones) that have built-in color. They include SuperCard, from



Silicon Beach Software (a subsidiary of Aldus Corporation), and PLUS, from Spinnaker Software. SuperCard not only offers color, but multiple windows, dialogs, and support for nearly all of the Macintosh interface. PLUS includes color and compatibility with IBM PC versions of PLUS stacks, in case you need to have stacks run under both the Mac and Windows environments. But if you need the special capabilities unique to HyperCard and it has to be in color, you must have this stack.





CHAPTER

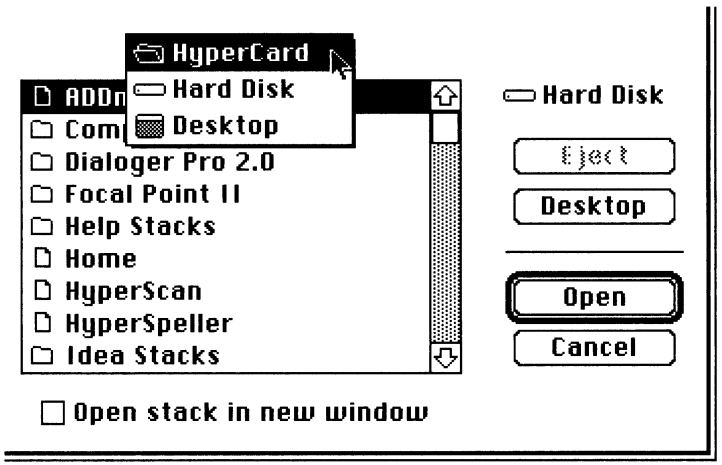
Stack Name:	**FullHPop**
Folder:	Building Blocks
Size:	30K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.1
Level of Expertise:	Very Advanced
Special Requirements:	UserLevel must be set to 5.

21

FullHPop: Everything's Popping Up Roses

One of the most common features of the Macintosh interface is missing from HyperCard. To see it while you are in HyperCard, select **Open...** from the **File** menu. Do you see the text in a box with a shadow near the top? Click and hold down the mouse when its cursor is positioned on the box. A menu drops down under your cursor. You can see this familiar detail in figure 21.1. This menu is known as a *pop-up menu*. Instead of dropping down from the menu bar, like the other menus, a pop-up menu can appear anywhere on the screen.

Figure 21.1
The pop-up menu in an open file dialog box.

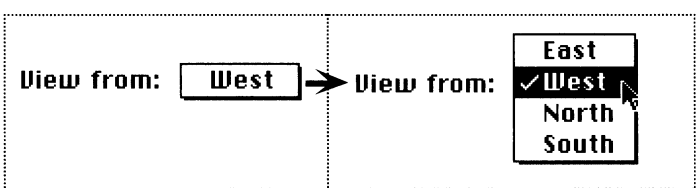


(By the way, this demonstration is over, so you can click **Cancel** to dispense with the open file dialog box.)

When to Use a Pop-Up Menu

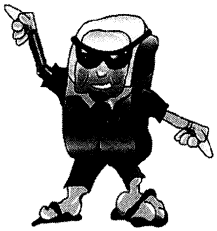
The most common place to use a pop-up menu is a place where you have to make a choice from a short list. Figure 21.2 is an example of a typical use of a pop-up menu. Imagine you have an application that lets you display an image of a building from four different vantage points: due east, west, north, and south.

Figure 21.2
A sample pop-up menu.



About Frédéric Rinaldi's Contributions

This stack is the first in a group of stacks from Frédéric Rinaldi, a developer with Apple France. M. Rinaldi is an incredibly prolific creator of external commands and functions for HyperCard. To date, he has whipped up 79 different externals. Some of them are in their third and fourth versions, with new features added all the time.



Many of M. Rinaldi's externals are immediately easy to use, but also include many optional features. This means that like the best Macintosh applications, the externals grow with you. You start with simple calls to the XCMD. After learning more about scripting with the XCMD, you might be ready to try something more ambitious. Then, you can add a new feature, because you have learned how to create the script for it.

How to Use This Stack

Open the FullHPop stack. You see a large scrolling field with information about this stack. To try out the XFCN, click and hold down the mouse when its cursor is over the shadowed button after the words **Menu list**.

As you can see, you can get some pretty complicated-looking pop-up menus with this XFCN. The good news is that most of the time you don't need to do anything this complex.

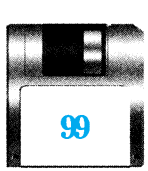
A Simple Example of FullHPop

To re-create the example you saw in figure 21.2:

1. Create a shadowed button somewhere on this card and name it `"West"`.
2. Open the script of the button and give it the following script:

```
on mouseDown
  put FullHPop("East,West,North,South") into aChoice
  if aChoice is not empty then set the name of me to item 1 of aChoice
end mouseDown
```

3. Save the script, close the script editing window, and choose the browse tool.
4. Click and hold the mouse button down over your button. You should see a pop-up menu just like the one in figure 21.2. If you do not, recheck your spelling, quotation marks, and parentheses. Notice that when you choose a different direction (like **South**), the name changes. When you click and hold again, the pop-up appears above, rather than below, with **South** checked. You get all of this standard behavior without having to include it in your script; FullHPop figures out that the checkmark should go next to the name of the button, and the menu should pop up at the right place. It's almost like magic.



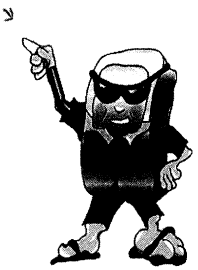
How to Use FullHPop in Your Stacks

To use FullHPop in your own stacks, copy the XFCN FullHPop from the resource fork of the stack FullHPop and paste it into the stack where you want to include a pop-up menu. Then create a button or field and give it a script like the example above. If you want to access more complex options in FullHPop, such as including icons or different fonts or hierarchical additions to the menus (those are the secondary menus that appear to hang off the original menu), check the information field on the FullHPop stack.

About System 7.0-Style Pop-Up Menus

Because some users felt that a shadow under the text was too subtle a cue that the text was actually a pop-up menu, Apple added another cue to this interface: a small arrow that is pointing down.

Since there is no button style in HyperCard that includes an arrow pointing down, you must include a little more scripting to get this part of the pop-up menu. In the FullHPop stack, there is a line of each FullHPop script that includes the words `ResizeMe`. This script resizes the shadowed button to account for the length of its new name, and repositions a small transparent button with the arrow icon to the right edge of the shadowed button. M. Rinaldi has already worked out the scripts to create this effect and they are in the FullHPop stack, so copy and paste them to your stack if you want to use System 7.0-style pop-up menus.



CHAPTER

Stack Name:	ListSelect
Folder:	Building Blocks
Size:	36K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.0
Level of Expertise:	Very Advanced
Special Requirements:	UserLevel must be set to 5.

22

ListSelect: A Stack for Freedom of Choice

This stack includes another external function (XFCN) from the prolific HyperCard developer Frédéric Rinaldi. ListSelect provides a standard Macintosh interface feature: a dialog box with a list in it. There are many options that you can use with the XCMD, but fortunately, to make a list appear, all you need to know are a few basic words.

Why Would I Want to Use ListSelect?

External functions like FullHPop andListSelect extend the interface of HyperCard stacks. They provide user input (in both cases, choices from lists) in forms that are already familiar to Macintosh users. Pop-up menus and lists in dialog boxes are old friends to us. In many cases it would be clumsy, crude, and (worst of all) un-Mac-like to ask the user to type something into a field. One of the caveats that all Macintosh programmers learn is "Never make someone type something if the computer already knows it." External functions like FullHPop and ListSelect make it possible to follow that recommendation.

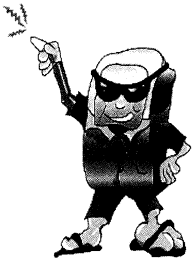
How to Use This Stack

Open the ListSelect stack. You see a field with some information about the stack. There are also some buttons and check boxes. Click the **ListSelect Test >>** button. You should see a dialog box with a list like figure 22.1.

Figure 22.1
A typical ListSelect dialog box.

Also, part of the script that created this particular version of the list dialog is placed in the message box. After you click the **OK** or **Cancel** button, you can hit return to see the dialog box again. This time you invoke the list dialog by sending the message that is in the message box. You can learn a great deal about the ListSelect XFCN by testing it this way:

1. Turn a check box on or off by clicking it. Each check box turns an option on or off for the ListSelect XFCN script.
2. Click the **ListSelect Test >>** button.
3. See the list dialog box, and notice how it has been altered.
4. Look at the contents of the message box. The text in the message box is the message that was used to show the list dialog box that you just saw. Press return to resend the message and take a look at the dialog box again.



Here's a neat trick: If you need to use many options with ListSelect, you can always copy the contents of the message box to use in scripts in your own stacks. Just drag over the text in the message box and press command-C or choose **Copy** from the **Edit** menu.

The **automatic demo** button shows many different variations of the list dialog box. If you want to see how all of them are done, take a look at the script of the **automatic demo** button.

All You Need to Know to Use ListSelect

To use ListSelect in your stack:

1. Copy the ListSelect XFCN from the resource fork of the ListSelect stack to your stack using ResEdit.
2. Create a button in your stack and give it the minimum script you need to get a ListSelect list. The script looks like this:

```
on mouseUp
  put ListSelect("1","Red,Orange,Yellow,Green,Blue") into myChoice
end mouseUp
```

This produces a dialog list box that looks like figure 22.2.

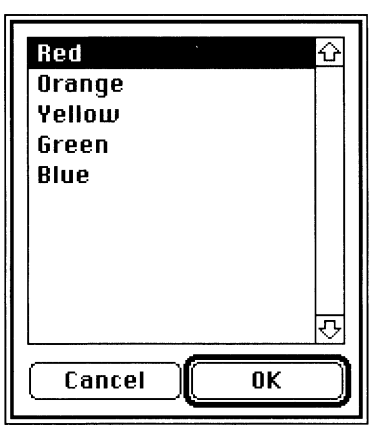


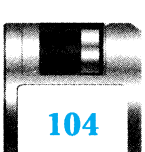
Figure 22.2
The simplest script for ListSelect produces a list dialog box that looks like this.

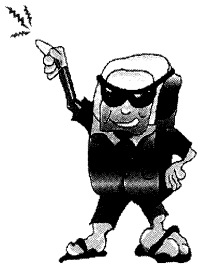
The number `"1"` in the script means that you can only choose one item at a time. Your list would probably contain different items from the colors in this example. You can use a field with the items seperated by returns or commas. Just use the field name in place of the words `Red`, `Orange`, `Yellow`, and so on.

The only other thing you need to know is what to do with the list. Since ListSelect is an XFCN, or external function, it returns a value to HyperCard. The value that ListSelect returns is what you chose from the list, or if you clicked the **Cancel** button. This value appears in the variable in the script called `myChoice`. It doesn't matter what you name your variable, as long as you choose a single word with no spaces or special characters like `#` or `&`. If you would like to see what `myChoice` contains, try inserting the line

```
put myChoice
```

after the line that includes `ListSelect` in the button script. The value of `myChoice` is shown in the message box after you choose a line in the dialog box. The command `put myChoice` is just to peek at what you get when you use ListSelect; you don't need to show the result in the message box in order to use it. Your own script could put the choice into a field, or make your script branch to a different task. That part is now up to you.





CHAPTER

Stack Name:	**Tabloid**
Folder:	Building Blocks
Size:	60K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.0
Level of Expertise:	Very Advanced
Special Requirements:	UserLevel must be set to 5.

23

Tabloid: No, This Is Not a Stack About *The Enquirer*

Have you ever wanted a spreadsheet or table in HyperCard? If you have, then you are in luck. This external command from Frédéric Rinaldi creates a new window in HyperCard that includes the cells for a spreadsheet-style table.

How to Use This Stack

To try out the Tabloid XCMD:

1. Open the Tabloid stack. You see a large scrolling field with information about this stack.
2. Click the button named **Tabloid**. A new window appears with spreadsheet-style cells in it. Figure 23.1 shows this window.

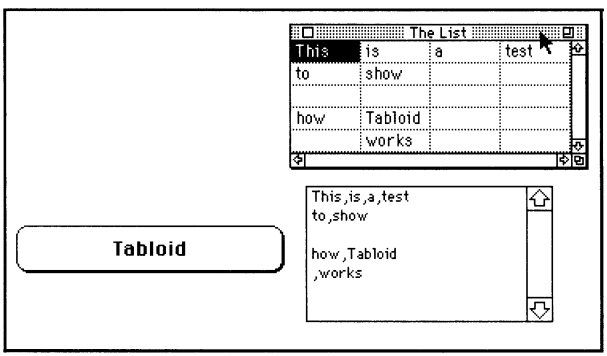


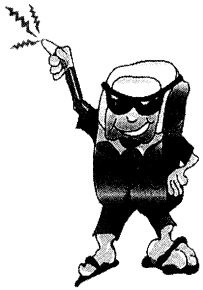
Figure 23.1
A spreadsheet window created with the Tabloid XCMD.

If you look closely, you'll notice that the words that appear in the cells of the spreadsheet window are the same words that appear in a field at the bottom right corner of the card.

3. To move the spreadsheet window around, click and drag the window's title bar. The title bar is thin, so this might be tricky at first.
4. Click any of the cells. The cell you clicked is highlighted. If you hold down the command key, instead of highlighting the cell, you can type new text into the cell. If you hold down the command key and click and drag near a cell border, you can resize the width or height of all the cells.

What Can You Do with Tabloid?

Tabloid gives you some new options for displaying information in HyperCard. A bunch of values would be much easier to read in a table-style Tabloid window, instead of a field separated by commas. If you work a little, you can even create a spreadsheet that calculates values in cells using HyperCard's built-in math functions.

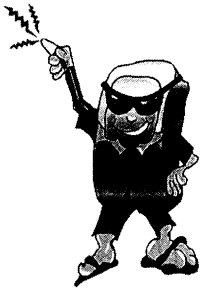


In addition, Tabloid knows when you click on a cell, and sends a message back to HyperCard containing the cell you clicked. You could take advantage of this message by creating a table of cards and going to the card you clicked in the table.

If you need a table, but not in a separate window, one of the options with Tabloid is to choose a specific window style. Some of the window styles include a standard Macintosh document window (with close box and zoom box) and round black windows like the calculator that Desk Accessory uses. Another style is a plain rectangular window, which appears to be on the card. By using a plain rectangular style and specifying its location before the Tabloid window appears (another option), you can simulate a table on the card itself.

What Tabloid Implies

HyperCard needs more ways of formatting information. There are five different styles of HyperCard fields: transparent, opaque, rectangle, shadow, and scrolling. These styles do not include a table or any kind of ruler or tab settings (or tabs) in the field. Tabloid fills a gap in HyperCard for this kind of field, but also includes the advantage of being able to exist as a seperate, floating window. Probably the most powerful floating window XCMD of all is Textoid, which follows in the next chapter.



CHAPTER

Stack Name:	**Textoid**
Folder:	Building Blocks
Size:	64K
Type of Mac:	Any Mac
RAM Installed:	2M
System Version:	6.05 or higher
HyperCard Version:	2.0
Level of Expertise:	Very Advanced
Special Requirements:	UserLevel must be set to 5.

24

Textoid: I Can't Believe It's an XCMD

Sometimes an external command can be so powerful and useful that you wonder how HyperCard ever got along without it. This XCMD is a prime candidate for "Most Valuable External." The Textoid XCMD gives HyperCard the ability to display windows of nearly any style with text in them. The text may be with or without styles, and the text and background may be in any color. You may specify that the text in the window is locked (read only) or unlocked so you can copy or change the text. You can transfer the text, complete with styles, to a field from the text window at

any time. Any word you click in the window can send a message back to HyperCard. Textoid provides, in essence, a new kind of field style for HyperCard: a field in a window!

How to Use This Stack

To try out Textoid, open the Textoid stack. You see a large field with information about the stack. Click the button at the bottom named **Textoid**. You see several windows appear with text in them. If you have a color monitor, you will notice that the text and background are in color.

Close each windoid by clicking in its close box. Watch closely and notice that the message box responds with **you closed window** and gives the name of the window you closed. This demonstrates that HyperCard can respond to a message that is sent from any textoid window when it is closed.

In case you are wondering what the difference is between a window and a windoid, the term *windoid* refers to any window on the Macintosh that uses a palette-style title bar. It is a miniature window, and hence a windoid. The term was invented by programmers, hence the technoid sound of it.

The Power of the Word

Textoid, like many of Frédérick Rinaldi's externals, is easy to start out with, because only three arguments or parameters are needed to begin. If you want to do more with Textoid, you can add extra parameters.

Try to call Textoid from the message box. Type the following into the message box:

```
Textoid "See","This is all it takes"
```

You should see something like figure 24.1:

Once your text window is up, you can do a lot with it:

1. While holding down the command key, click in the content area of your windoid named See. The text unlocks.
2. Select the word *takes*.
3. Choose **Bold** from the **Style** Menu. The word *takes* becomes bold.

4. Click between the words *all* and *it*. Choose **Monaco** from the **Font** menu. Type the word `that`. Your typing appears as Monaco. As you can see, the textoid behaves just like a HyperCard text field. **Copy** and **Paste** work too. Copy the first sentence. Now click at the end. Choose **Paste** five or six times. As the window fills up with text, a scroll bar appears to the right, again, just like a scrolling field.

5. Hold down the option key with the cursor over the textoid window. The cursor becomes a *grabber hand*. You can hold the mouse button down and drag to move the text up or down within the window.

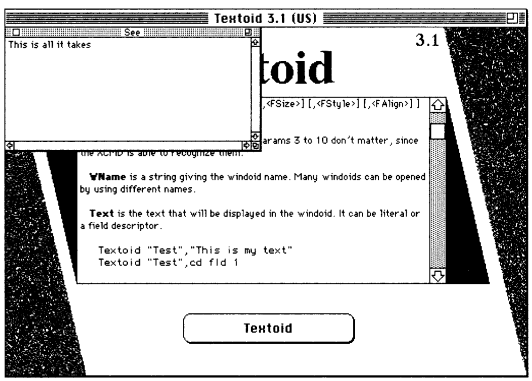

Textoid 3.1 (US)
See
This is all it takes
3.1
toid
,<FSize>][,<FStyle>][,<FAlign>]]
arams 3 to 10 don't matter, since
WName is a string giving the windoid name. Many windoids can be opened by using different names.
Text is the text that will be displayed in the windoid. It can be literal or a field descriptor.
Textoid "Test","This is my text"
Textoid "Test",cd fld 1
Textoid

Figure 24.1
A call to Textoid including only the window title and text to display.

Other Windows

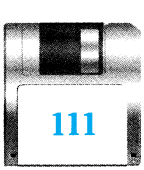
Close your first window.

If you want to get still more adventuresome with Textoid, try typing this in the message box:

```
Textoid "Documentation", "cd fld 1","plain","4,20,450,300"
```

In a moment you see a standard Macintosh window holding the entire text of the field about Textoid, complete with all of its styles.

There are many different types of windows that Textoid produces. You can create as many as memory allows. Figure 24.2 shows a few of them at one time.



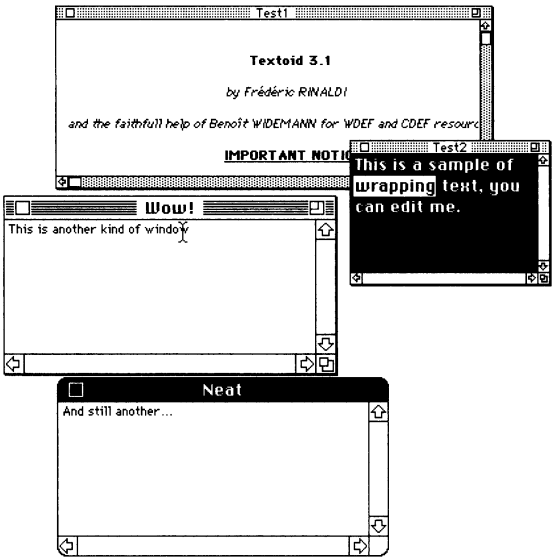
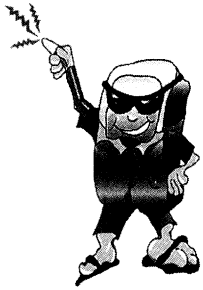


Figure 24.2
Several text windows all created by Textoid.

Closing Windows and Clicking Text in Windows

When you close a Textoid window, you can notice that it is closed in a handler that intercepts the `closeTextoid` message. You can transfer the text of the textoid to a holding field somewhere, with its style intact. Before a window is shown, you may also place the insertion point anywhere in the window. This might be handy for creating a note-taking system that always places the insertion point at the end of the text, ready for you to continue adding to your notes.

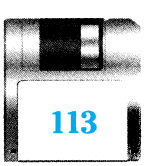
Textoid can send messages about words you click when the text in a window is locked. Imagine a foreign language text that, first, used HyperCard's Find command to find the word you clicked in the window and, second, showed the word by looking it up in a HyperCard stack of words and their translations. HyperCard could even give the correct pronunciation of the word with digitized voice. The possibilities are endless.

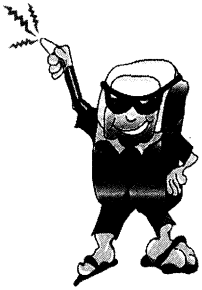


Closing Thoughts

This stack completes this collection. Chapter 25 is entitled "Cool Solutions to Uncool Problems." It is a troubleshooting section for common problems found by stack browsers and beginning HyperCard authors. Chapter 26 provides information about contacting on-line services and user groups that can serve as a HyperCard resource.

If nothing else, *Cool Mac Stacks* shows the variety of the creations possible with this software. HyperCard is waiting for you to discover it. I wish you good luck and happy stacking.





CHAPTER

25

Cool Solutions to Uncool Problems

This section offers some quick fixes to common problems with HyperCard.

Problem:

When I try to open one of the stacks in the *Cool Mac Stacks* collection, I get the following message: "New File Format requires new version of HyperCard."

Solution:

Your version of HyperCard is probably version 1.0, 1.1, or 1.2.5—in other words, HyperCard 1. The stacks in this collection are in the newer HyperCard 2 format. To open these stacks, you need to get a version of HyperCard that is 2.0 or greater (the most recent version is 2.1, which is required for a few of the stacks in this collection). You can get HyperCard 2.1 with a System 7 upgrade from Apple. You can also get it from your dealer or from Claris Corporation. HyperCard is no longer a free product, but fortunately it is not very expensive, (usually about sixty dollars).

Problem:

When I try to open HyperCard, I get a message that looks something like this: "There is not enough memory to open 'HyperCard' (1,000K needed, 999K available). Do you want to open it using the available memory?"

Solution:

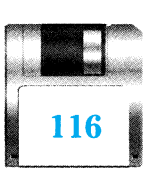
This is one of those really annoying messages that the Mac gives you, such as "Application Busy or Missing" (What do you mean it's busy? Does it have a hot date?). When you are asked if you want to open HyperCard anyway, the OK button should really say "Of course, you stupid machine!" If the amount of available memory is a lot lower than the amount needed (if the amount available was 600K, for example), you would probably not want to try to open HyperCard using the available memory. HyperCard needs at least one megabyte (or 999K) to run at all, and for all practical purposes, it needs two megabytes (1999K) to run swiftly and comfortably. For more information about HyperCard's memory requirements, see Chapter 1, "About This Cool Book."

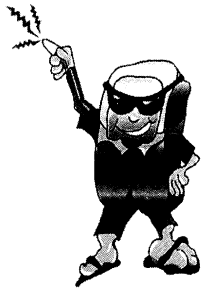
Problem:

When I try to open HyperCard, I get a message that looks something like this: "These font(s) are missing from your system: Palatino10, 12, 14, 18, 24. For text to display correctly in the stacks make sure you have installed the fonts included with HyperCard."

Solution:

HyperCard 2.0 comes with a set of screen fonts, including Courier, sizes 10 and 12, Geneva, sizes 10 and 14, Palatino, sizes 10, 12, 14, 18, and 24, and Helvetica, sizes 18 and 24. Install these fonts in System 6 using the Font/DA Mover, or in System 7 by dragging them directly into your system file in the system folder.





This message is annoying, but it is not all that dangerous (unless your sense of aesthetics makes it a major issue).

Problem:

My menu bar is missing the **Tools** or **Objects** menus.

Solution:

Your level of access to HyperCard has been limited to a level called *browsing* (level 1) or *typing* (level 2). If you would like to change your level of access, called userLevel, do the following:

1. Open HyperCard and go to the Home stack. If you start HyperCard by double-clicking the HyperCard icon in the Finder, you will already be in the Home stack.
2. Go to the User Preferences card of the Home stack. It is the last card in the Home stack, so choose **Last** from the **Go** menu, or press command-4.

If you are missing some text on the right side of the screen and numbers 3-5 on the left, then you have a copy of HyperCard Lite (the special Home stack shipped with the System 7.0 upgrade, as well as new Macintoshes), and there is an intermediate task that you must perform:

HyperCard Lite users must make some features on this card visible by typing a secret word. Choose **Message** from the **Go** menu, and type `magic` into the message box. Press return. The text and numbers appear.

3. Change your userLevel by clicking a higher number than 1 or 2. My advice is to click 5 and forget about it. Don't worry, you can handle it.

Problem:

I can't see the menu bar.

Solution:

Press command-spacebar. This toggles the menu bar on and off.

Problem:

When I open some stacks, I see part of the card, and I also see a strange box on the screen.

Solution:

Rather than use the standard Macintosh interface of scroll bars on the edge of the document window for scrolling around a window, HyperCard uses a unique control for moving around the card called a *scroll windoid*. A windoid is a special window that always remains in front of other windows. The message box and tool palettes of HyperCard are actually windoids. The scroll windoid is a miniature representation of the card window, with the visible portion in a smaller rectangle within the windoid. To move the visible area on the card, click and drag the rectangle within the scroll windoid.

To resize the card window temporarily, position your cursor near the edges of the smaller rectangle within the scroll windoid. The cursor should change to signal that you may now resize the window. Click and drag to temporarily resize the card window.

Problem:

HyperCard keeps telling me "Not enough memory to use the paint tools."

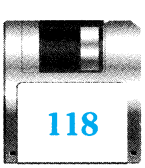
Solution:

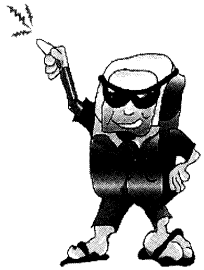
If you are running HyperCard under MultiFinder (or System 7.0), try allocating more memory to the HyperCard application. To do this, while in the Finder desktop, click the HyperCard application icon once to select the icon. Then choose **Get Info...** from the **File** menu. You see the Info For... window for HyperCard. Type a number higher than what you see in the box at the bottom and close the window. The recommended allocation for HyperCard is1000K, but the application runs fastest and is most stable when there is more than 2000K of RAM allocated. Running HyperCard when there is less than1000K available is possible, but HyperCard doesn't have enough memory to use the paint tools in a low-memory situation.

If you are running under System 6.05 (or any other System 6 version that HyperCard 2 may run under, such as 6.07 or 6.08), try running HyperCard in the Finder, rather than MultiFinder, because this also frees up some RAM.

If you still get this low memory message, your system could be cluttered up with lots of memory-hungry additions, like INITS, control panels, and other system extensions.

The way you know you have system extensions is to look for icons to appear at the bottom of your screen as your Mac starts up. The rule of thumb with system extensions is, "If it just makes things look prettier, you can probably live without it." Remove any unneccessary extensions from the System Folder (or under System 7.0, the Extensions Folder) and restart your Macintosh. This also frees up some more memory that may be allocated to HyperCard.





The last resort is to buy more memory chips, or *SIMMS*, as they are often called. The good news here is that it keeps getting cheaper and easier to add extra RAM to your Macintosh. Your Mac remains compatible with all of your software no matter how much RAM you add. In addition, you can open more and larger documents, run more applications at once, and indulge in more silly (and useful) system extensions.

Problem:

I click a button and nothing happens, even though it worked a minute ago.

Solution:

Make sure you are using the browse tool. If you don't see the browse cursor, then you are an author, not a user, of HyperCard. Become a user by choosing the browse tool or pressing command-tab.

Problem:

I had some nice background art, but I dumped paint all over the card and can't see it any more.

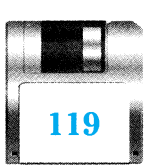
Solution:

It's easy to make a mistake with the paint bucket tool. Clicking with the paint bucket tool cursor dumps the currently chosen pattern all over the screen where there is an area enclosed by a black continuous border, or there is a continuous area of black pixels.

If neither of these graphic objects are on the screen, paint spills everywhere, all the way to the edges of the card window. Unfortunately, the default pattern for paint in HyperCard is solid black, so an unintended click can really produce a dramatic blackout!

To remove all paint from either the card or background layer:

1. Make sure you are editing the layer you want to remove paint from (probably the card layer). If you are in the background layer and want to move to the card layer, be sure to choose **Background** from the **Edit** menu to leave the background layer.
2. Drag the **Tools** menu off of the menu bar to turn it into a tool palette.
3. Double-click the eraser tool. All paint in the current layer disappears.



Remember that HyperCard saves everything you create or change, so make sure that you have your fingers poised over command-Z (keyboard equivalent for **Undo**) when using the paint bucket tool.

Problem:

I can't find text that I enter into fields. I also can't seem to change text that I type into fields.

Solution:

HyperCard's interface can cause some confusion over which tool to use to enter text. The browse tool (with a finger) is used to click buttons and enter text in fields. The tool that looks like an "A" in the tool palette is used to create paint text. Paint text consists of dots on the screen that only look like text, not actual intelligible text. You often use paint text to decorate cards and backgrounds and provide labels for the screen. If you type paint text using the paint text tool on top of a field, it will appear as if you are entering text into the field, but you are merely painting letters on top of the field. You may notice that as you type real text, the text wraps to the edges of the field. Paint text doen't stay within the confines of the field. Trying to find the paint text is asking HyperCard to perform optical character recognition on all the paint in your stack, to determine what looks like the text you are searching for. Maybe HyperCard Version 9 on a Mac VI will be able to do this, but not this version. One advantage of paint text is that if you want to display some decorative text (not to be searched for or changed later), you don't have to worry about whether the browser of your stack has the unusual font you use to create the lettering on their Mac. The decorative text is just paint.

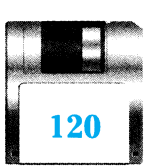
Problem:

I want some text to appear on every card. Do I have to enter it every time I create a new card?

Solution:

Sometimes you need a label for a field that should appear on every card. There are two solutions for this in HyperCard 2.0. One was just mentioned: painted text. To make painted text appear on every card for a background:

1. Choose **Background** from the **Edit** menu. You see the hatch marks on the menu bar showing that you are editing the background.
2. Choose the painted text tool, the one that is a letter A.



3. Click where you would like your text. Start typing. You cannot change this text after you stop typing and click somewhere else on the screen, so try and get it right the first time. You can always erase the text you type with the eraser tool and try again.

4. Choose **Background** from the **Edit** menu and choose the browse tool.

The other solution for text on every card is a new feature in HyperCard 2.0 called *shared text*. Shared text is a *property* of a field. Other field properties include whether the field has regular or wide margins, whether or not lines are showing, or whether you may type text into the field or not (an unlocked field is editable, a locked one is not). Shared text is a property that only may be turned on for background fields. To make text appear on multiple cards using the shared text property:

1. Choose **Background** from the **Edit** menu and create a new field. If you already have a background field in mind for this shared text, you may use the field you already created.

2. Double-click the field to see the field info dialog box. Turn on the shared text property by clicking the check box named **Shared Text**. Notice that specifying **Shared Text** also checks the **Don't Search** property as well. Click the **OK** button.

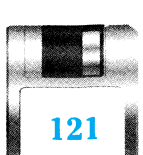
3. Choose the browse tool. Now, while still in the background layer, type into the field the text that you would like to see on every card. You can change the font and size just as you normally would.

4. When you are done, choose **Background** from the **Edit** menu. The text in the field with shared text is no longer editable, but it does show up on every card in this background. If you need to make changes to the shared text, you must choose **Background** from the **Edit** menu in order to do so.

The major advantage to using shared text is that when you print the card, the text prints at the resolution of your printer. You can produce excellent printed forms by using fields with shared text on a HyperCard card set to the size of a sheet of paper on a laser printer. All text on your form, whether it is information that has been filled out, or labels for the form, prints out at high resolution. The disadvantage to using shared text is that if the recipient of your stack doesn't have the same fonts that you do, text in the shared text fields does not appear in the same fonts on their Macintosh as it does on yours.

Problem:

I think this stack has some invisible buttons, and I'd like to see where they are.



Solution:

You can peek at invisible (transparent) HyperCard buttons on both the card and background by holding down the option and command keys at the same time.

If you click one of the buttons that you see with the option and command keys held down, you open up the script of the buttons and can look at or edit the script.

Problem:

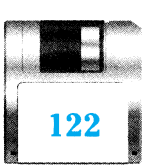
When I click on icons on my Home stack that open other stacks, I get an open file dialog box saying "Where is (the name of my stack)?"

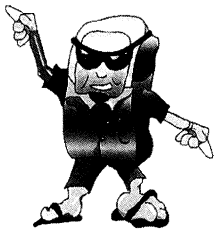
Solution:

You might be missing some of the stacks that this Home stack is expecting to find on your hard disk. If you have other disks of HyperCard stacks, be sure to copy the contents of them to your hard disk too. You might see the Where is... file dialog box one more time. After you show HyperCard where the stacks it is looking for really are, however, HyperCard takes note of any folders or hard disks in which it can look for stacks before it bothers you again. If you never expect to use a stack that is linked to your Home stack after you have looked at it once (like the HyperCard Tour, that appears on some copies), feel free to throw the stack document in the trash. You may delete the button on the Home card as well. To do this:

1. Choose the button tool from the **Tools** menu. The button tool is covered in the tutorial.
2. Click the button you want to delete
3. Choose **Clear** from the **Edit** menu, or press the delete key. The button disappears.
4. If this is a transparent button placed over paint, use the eraser tool to erase the paint picture as well.
5. Choose the browse tool.

Deleting the button that links to a stack does not remove the stack itself. Delete stacks by dragging the stack document icon to the trash in the Finder desktop, or by opening the stack you want to delete in HyperCard and choosing **Delete Stack...** from the **File** menu.





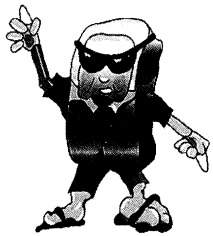
The paths of folders where HyperCard stores freqently searched folders and hard disk names are in fields on three cards in the Home stack. There is a field containing paths to folders frequently searched from stacks, one for applications that you may launch from HyperCard, and one for documents that you may open with the applications you launch from HyperCard.

Problem:

A script seems to be caught in an infinite loop. How can I stop the script and regain control of my Mac without pulling the plug?

Solution:

HyperCard can get stuck repeating something forever, so remember that you can cancel nearly any script in process by pressing command-period. The only time you can't abort a script with command-period is when an author has blocked this capability by setting the `cantAbort` property of this stack to `true`. (The `cantAbort` property is only available beginning with HyperCard 2.) If this has been done, hopefully the author really knows what he or she is doing. If not, then yes, hit the reset button on the side of your Mac (or front, depending on the model you have). In five years, I have never seen a script with an infinite loop with no abort allowed, so hopefully it won't happen to you.



26

On-Line Services and User Groups

Electronic Telecommunications Services

CompuServe, Inc.
5000 Arlington Centre Boulevard, P.O. Box 20212, Columbus, OH 43220
(800) 848-8990

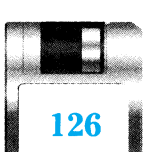
GEnie
General Electric Information Services Company
401 N. Washington Street, Rockville, MD 20850
(800) 638-9636

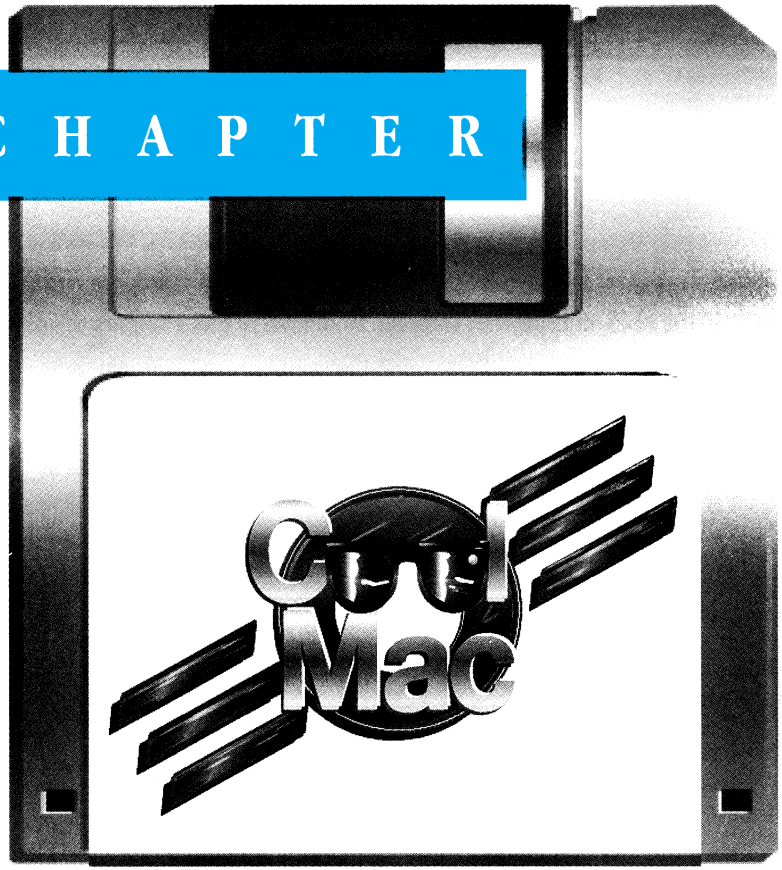
American Online
Quantum Computer Services, Inc.
Customer Relations Department at (800) 227-6364

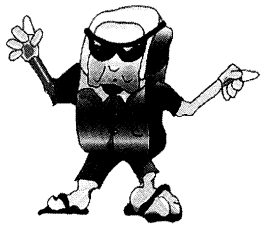
User Groups

The Boston Computer Society Macintosh Group (BCS•Mac)
48 Grove Street, Somerville, MA 02144
(617) 625-7080

Berkeley Macintosh User's Group (BMUG)
1442A Walnut St. #62, Berkeley, CA 97409-1496
(415) 549-BMUG







27

Browsing Quick Reference

To open HyperCard, double-click the HyperCard icon on the Finder desktop, or double-click a HyperCard document icon.

To navigate to the first card in a stack, choose **First** from the **Go** menu, or press command-1.

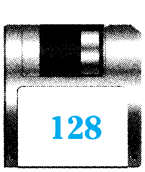
To navigate to the next card in a stack, choose **Next** from the **Go** menu, or press command-2.

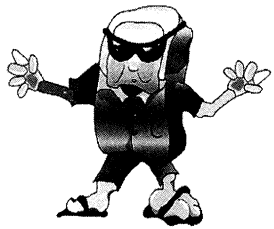
To navigate to the previous card in a stack, choose **Previous** from the **Go** menu, or press command-3.

To navigate to the last card in a stack, choose **Last** from the **Go** menu, or press command-4.

To look at miniature pictures of cards you have seen recently and to navigate to one of the cards, choose **Recent** from the **Go** menu and click the picture of the card you would like to revisit. **Recent** is also available from the keyboard by pressing command-R.

To find text in the stack, choose **Find** from the **Go** menu, or press command-F. Type the text you wish to find and press return.





Index

3-D animation, 79
4-D images, 61

A

abacus, 42
About Boxer stack, 89-92
About This Stack
  button, 94
access levels, 117
Addresses with Audio
  icon, 12
America Online, 48, 126
anagrams, 39
animation, 62
  3-D, 79
  PICS, 79
  play, 82
  repeating, 83
AppleEvents, 68
Appointments with Audio
  icon, 12
arguments, 28
ARS MAGNA, 39
art, 62
ASCII format, 80
astronomical events, 47
authoring mode, 7
automatic demo
  button, 103

B

Back command, 14
backdrops, color, 94
background, 8-9
  buttons, 22-24
  editing, 19-20
  field, 22
  paint, removing, 119
Background command,
  119-120
Balloons stack, 69-72
BCS•MAC (Boston Com-
  puter Society Macin-
  tosh Group), 126
Berkeley Macintosh
  User's Group
  (BMUG), 126
blackouts, 119
BMUG, 126
Bond of Union stack,
  61-62
borders, continuous, 119
Boston Computer Society
  Macintosh Group
  (BCS•Mac), 126
Broadcast, 68
browse
  cursor, 119
  mode, 7
  tool, 8, 25, 119
browsing, 11-16
  access level, 117
  quick reference, 127
  recipes, 33
buttons
  About This Stack, 94
  automatic demo, 103
  background, 22-24
  Convert, 38
  creating, 91
  invisible, 121-122
  New, 19
  SEND, 66
  Stack Overview, 94
  Test>>, 76
  Text About Box, 91
  Today, 46
  ZONE, 66

C

calculating
  astronomical events, 47
  sunrise time, 45
  twilight time, 45
card layer, 8
cards
  miniature, 128
  navigating, 127
  text, repeating, 120
  User Preferences, 117
changing text, 15
characters, optical
  recognition, 120
Chicago font, 90
Clear command, 122
clipboard, copying
  scripts, 91
closeTextoid
  message, 112
CODE resources, 27

coincidence generators, 39
color, 80, 93
  backdrops, 94
  monitors, 94
  pictures, 95
Colorizing HyperCard stack, 93-95
commands
  Back, 14
  Background, 119-120
  Clear, 122
  Copy, 111
  Delete Stack, 122
  Export Recipe, 34
  external, see XCMDs
  Find, 15, 128
  First, 127
  Get Info, 118
  Import Recipe, 34
  Last, 117, 128
  Lock Screen, 86
  New User, 64
  Next, 127
  Open, 13
  Paste, 111
  playPICS, 79
  Previous, 127
  Print Recipe, 34
  Recent, 14, 128
  Show Balloons, 69
  Show Index, 33
  Sort Recipes, 34
  Stack Info, 77
  start using, 76
  Undo, 120
compile, 10
CompuServe, 125
conferencing, 63-64
connecting to other Macs, 66
Convert button, 38
coordinates, rectangle, 95
Copy command, 111
copying
  scripts to clipboard, 91
  stacks, 122
creating
  buttons, 91
  dialog boxes, 89-90
  graphics, 19-20
  scripts, 89
  stacks, 8, 17-18
  status window, 87
cursor
  browse, 119
  grabber hand, 111
  I-beam, 7
  pointing finger, 7

D

data fork, 27
Delete Stack command, 122
deleting stacks, 122
dialog boxes
  creating, 89-90
  save, 18
  "Where is (the name of my stack)?", 122
dialogs, 67
displaying color, 94
documents, 7
Don't Search, 121

E

Edit menu, 119
editing
  background, 19-20
  text, 15
Eighth Book of Tan, The, 53
eraser tool, 122
Escher, M.C., 61
Export Recipe command, 34
exporting recipes, 32-34
extensions, system, 118
external commands, see XCMDs
external functions, see XFCNs

F

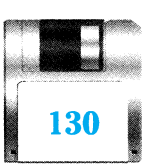
fields
  background, 22
  naming, 22
  opaque, 107
  property, 121
  rectangle, 107
  resizing, 22
  scrolling, 107
  shadow, 107
  text, 15, 120
  Transcript, 67
  transparent, 107
File menu, 64
files, PICS, 80
Find command, 15, 128
Finder, 118
First command, 127
fonts, 90, 111
Fonts menu, 111
forks, 27
formats
  ASCII, 80
  TIFF (Tagged Image File Format), 80
freeware, 4
front end, 47
FullHPop stack, 97-99
functions
  external, see XFCNs
  math, 106

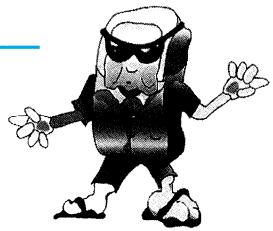
G

games, 49-56
geeks, 10
GEnie, 126
Get Info command, 118
Go menu, 13, 117
grabber hand, 111
graphics, creating, 19-20

H

handlers, 91
  mouseEnter, 72
  mouseLeave, 72
  mouseUp, 72





hatch marks, 19-20, 120
help, Balloon, 69
Home stack, 12
HyperCard, 2-4
  navigating, 14
  stacks, sharing, 3
  starting, 12
HyperConference, 63-68
HyperTalk, adding words, 28
hypertext, 6

I

I-beam cursor, 7
Import Recipe command, 34
importing recipes, 32-34
insertion point, 112
installing disk, 133
invisible buttons, 121-122

L

Last command, 117, 128
latitude, 46
ListSelect stack, 101-104
Lock Screen command, 86
locking and unlocking text, 110
longitude, 46
loops, scripts, 123

M

Macintosh programming, 10
math functions, 106
Memex (memory extender), 6
memory
  chips, 119
  not enough, 118
  read-only, 10
menu bar, 32, 117
menus
  Edit, 119
  File, 64
  Fonts, 111
  Go, 13, 117
  hatch marks, 120
  Objects, 22, 77, 117
  Patterns, 20
  pop-up, 97-100
  Recipes, 32-33
  Rman, 51
  Tools, 7, 117
message box, 15
messages
  AppleEvents, 68
  closeTextoid, 112
  Not enough memory to use the paint tools, 118
  responses, 72
microfilm, 6
modes, 7
monitors, color, 94
MultiFinder, 118

N

naming
  fields, 22
  stacks, 18
networks, 63-67
  Broadcast, 68
  HyperConference, 63-68
  passwords, 64
  users, 64
New button, 19
New User command, 64
Next command, 127

O

Objects menu, 22, 77, 117
on-line services and groups, 125
opaque fields, 107
Open command, 13
opening
  HyperCard, 127
  stacks, 13
optical character recognition, 120

P

paint, 119-120
paint bucket, 20
paint tools, 19, 118
palette style title bar,
parameters, 28
passwords, 64
Paste command, 111
pasting scripts, 76
patterns, 93
Patterns menu, 20
Phone->Acronym stack, 37-38
PICS
  animation, 79, 82
  file, 80
PICS Player stack, 79-83
PICT frames, 80
pictures, color, 95
pixels, black, 119
Pizza Navigator stack, 57-60
playPICS command, 79
pointing finger cursor, 7
pop-up menus, 97-100
Previous command, 127
Print Recipe command, 34
problems, *see* troubleshooting
Program Linking, 64
programming Macintosh, 10
property of fields, 121
puzzles, 54

R

read-only memory (ROM), 10
Recent command, 14, 128
recipes, 31-34
Recipes menu, 32
Recipes stack, 31-32
Rect, 95
rectangle fields, 107
rectangles, 95
reference, browsing, 127
Renaissance Man stack, 49-50
Resource Fork, 27, 81
resources, 27, 125
Rman menu, 51
ROMs, 10

S

save dialog box, 18
Say the Numbers stack, 75-78
screens
  locking, 86
  Theatre, 80
scripts, 76
  aborting, 123
  animation, 82
  copying to clipboard, 91
  creating, 89
  in loops, 123
  multi-stack, 76
  pasting, 76
scroll bars, 118
scroll windoid, 118
scrolling fields, 107
searching, 34-35
secret word, 117
SEND button, 66
shadow fields, 107
shared text, 121
shareware, 4
shopping list, 34
Show Balloons command, 69
Show Index command, 33
SIMMS, 119
Sort Recipes command, 34
sorting, 32
sound, 67
speaking, 75
spreadsheets, 105
Stack Info command, 77
Stack Overview button, 94
stacks, 4-6
  About Boxer, 89-92
  background, 8
  Balloons, 69-72
  Bond of Union, 61-62
  Colorizing HyperCard, 93-95
  copying, 122
  creating, 8, 17-18
  deleting, 122
  foreground, 8
  FullHPop, 97-99
  Home, 12
  ListSelect, 101-104
  missing, 122
  naming, 18
  navigating, 13
  opening, 13
  Phone->Acronym, 37-38
  PICS Player, 79-83
  Pizza Navigator, 57-60
  Recipes, 31-32
  removing, 122
  Renaissance Man, 49-50
  Say the Numbers, 75-78
  sharing, 3
  Status Window, 85-88
  Sunrise Demo, 45-46
  Tabloid, 105-107
  Tangrams, 53-56
  Textoid, 109-112
  WildAbacus, 41-42
stackware, 16
start using command, 76
starting HyperCard, 12
status window, creating, 87
Status Window stack, 85-88
Sunrise Demo stack, 45-46
sunrise time, 45
system extensions, 118

T

tables, spreadsheet-style, 105
Tabloid stack, 105-106
Tangrams stack, 53-56
Tans, 53-56
Test>> button, 76
text
  changing, 15
  entering, 15, 120
  fields, 15, 120
  finding, 15, 128
  locked, 109
  multiple cards, 120
  paint, 120
  shared, 121
  unlocking, 110
  wrapping, 120
Text About Box button, 91
Textoid stack, 109-112
Theatre screen, 80
TIFF (Tagged Image File Format) format, 80
time zone, 45
title bar, palette style, 110
Today button, 46
Toolbox, 10
tools
  browse, 8, 25, 119
  eraser, 122
  paint, 19, 118
Tools menu, 7, 117
Tools palette, 7, 20
Transcript field, 67
transparent fields, 107
troubleshooting, 115-123
twilight time, 45
typing access level, 117

U

Undo command, 120
User Preferences card, 117
userLevel, changing, 117
using the disk, 133

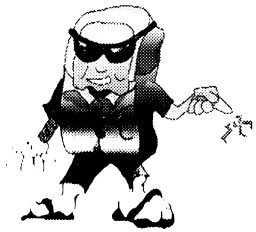
W

"Where is (the name of my stack)?" dialog box, 122
WildAbacus stack, 41-42
windoids, 110, 118
windows
  resizing, 118
  styles, 107
words, adding to HyperTalk, 28
wrapping text, 120

X–Z

XCMDs, 27-28, 109
XFCNs, 28, 71-73

ZONE button, 66



Using the Cool Disk

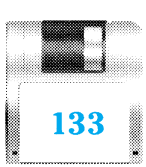
You will need a hard disk with at least one and a half megabytes (1500K) of free space to install the software that comes on this disk. To prepare the disk for use, lock the disk (by sliding the black plastic shutter on the disk to the open position) to protect the contents of the disk from any careless Mac actions you may take in the near future. To install the *Cool Mac Stacks* disk, follow these steps:

1. Insert the locked disk into your floppy drive.
2. Double-click on the icon of the floppy disk.
3. Double-click on the icon of the compressed file **Cool Mac Stack.sea**.
4. When the prompt appears asking where you want the uncompressed files, choose to install them on your hard disk or in a folder on your hard disk.
5. Click the **Extract** button when you are satisfied where the folder will be placed. The files decompress automatically.
6. Slide the icon of the floppy disk to the trash to eject the disk.
7. Double-click on the folder names of the newly decompressed files.

That's all there is to it. Enjoy!

How This Book Was Created

This book is especially cool because it (along with all Hayden titles) was created entirely on a computer. The text was written in Nisus and edited using Microsoft Word. Pages were laid out using PageMaker. The index was created via PageMaker. The cover was designed using Adobe PhotoShop and Quark XPress and icons and other images were created using Adobe Illustrator and FreeHand. Even all correspondence—from the initial book contract to "the manuscript looks great" and "where is my check?" letters—were created in Microsoft Word on the Mac. Pretty neat, huh?



What's on the Cool Disk?

The disk contains lots of cool stacks—19 to be exact. Uncompressed, the stacks take up over one megabyte of hard disk space. The stacks cover all the latest HyperCard gimmicks, from System 7 Program Linking (AppleEvents) to color to *mucho cool* animation.

There are four categories of stacks: Fun and Games, Utilities, Building Blocks, and Information Enhancement. Following is a short description of each stack on the disk:

Information Management:

Recipes is an electronic recipe file—the sort of high-tech card file that HyperCard was originally created for.

Phone->Acronym generates letter equivalents for your phone number. Call me, I'm "FUN 4YOU".

WildAbacus lets you play with an abacus, whether you're just learning how to use it or you showing off your ancient mathematical skills.

Sunrise Demo tells you the time of the sunrise and sunset on any date for any location within the continental United States.

Fun and Games:

Renaissance Man is the original Hangman game—in boxer shorts.

Tangram is a computerized version of the ancient game of forming pictures from a set of different geometric shapes.

The Pizza Navigator is a what-if stack which lets you try out different pizza combinations, assess the cost differences, compare local pizza parlors, and it dials the phone number.

Bond of Union is an animated version of the original drawing by M.C. Escher, turning a 3-D image into a 4-D image.

Utilities:

HyperConference is a stack specifically designed to take advantage of System 7's real-time conferencing capability.

Balloons is a System 7 utility stack which enables you to create customized Balloon Help.

Say the Numbers is a stack which uses a digitized voice to speak any numbers you type in.

Building Blocks:

PICS Player enables you to animate your HyperCard stacks.

Status Window is a building block you can use to show status report messages to users of your stacks.

About Boxer is a building block used to create About Boxes (you know, under the Apple menu).

Colorizing HyperCard adds color to your HyperCard stacks.

FullHPop enables you to create pop-up menus in HyperCard.

ListSelect enables you to choose one or many items within a list appearing in an on-screen dialog box.

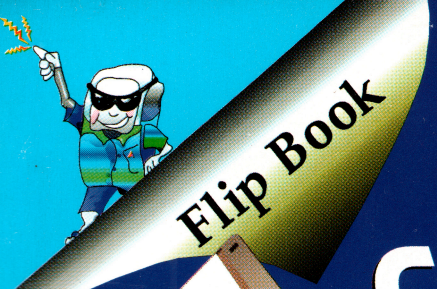
Tabloid displays windows with spreadsheet-style cells and tables.

Textoid gives Hypercard the capability to display windows of nearly any style with text in them.

Copyright © 1992 Hayden and its licensors. All rights reserved.
By opening this package you are agreeing to be bound by the following agreement.

This software product is copyrighted, and all rights are reserved by the publisher, its licensors, and author. TeachText may not be modified, networked, rented, leased, loaned, or distributed in whole or in part. You may not electronically transmit TeachText from one computer to another or over a network. You are licensed to use this software on a single computer. Making copies of the software for any purpose, other than that needed to facilitate its use on a single computer, is a violation of the United States copyright laws.

This software is sold AS IS without warranty of any kind, either expressed or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Neither the publisher or its licensor(s), nor its dealers or distributors assumes any liability for any alleged or actual damages arising from the use of this program. (Some states do not allow the exclusion of implied warranties, so the exclusion may not apply to you.)



Cool Mac™ Stacks

INFORMATION CENTER

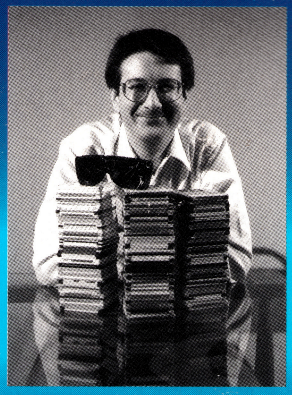
- All Skill Levels
- Book / Disk
- HyperCard 2.1 & 2.0
- General Macintosh

What you get for your money

- A simple explanation of how to use HyperCard
- Directions for using the HyperCard stacks on the disk
- Ideas for creating your own stacks
- Instructions for modifying the stacks provided
- 19 fun, ready-to-use stacks, including...
  - -Pizza Navigator
  - -Renaissance Man
  - -Colorizing HyperCard
  - -Say the Numbers
  - -Balloon Help

...and many more!

See inside for details!



David Drucker is Director of Education for the Boston Computer Society Macintosh group (BCS•Mac). He led the Hypercard Multimedia group at BCS•Mac and has since worked as a consultant. David is now a Macintosh trainer and developer and, of course, he is very cool!

$19.95 USA
$24.95 CAN

HAYDEN

"One of the best discussions I've seen on Hypercard...fun reading too!"
–Mel Beckman
President
Beckman Software Engineering

"Hallelujah!"
–George Fredericks of Handel

"This will be bigger than the Beatles."
–Tad Ringo

ISBN 0-672-48522-2

9 780672 485220 90000